Cross-Layering et routage dans un réseau ad hoc : politique de relais de trame sur un réseau de capteurs sans fil organisé selon une topologie en arbre

Nancy El Rachkidy

N° d'ordre : D. U. 2195

E D S P I C : 54

Université Blaise Pascal - Clermont II

École Doctorale
Sciences Pour l'Ingénieur de Clermont-Ferrand

Thèse

Présentée par

Nancy El Rachkidy

pour obtenir le grade de

Docteur d'Université

SPÉCIALITÉ : Informatique

Cross-layering et routage dans un réseau ad hoc :
Politique de relais de trame sur un réseau de capteurs sans fil organisé selon une topologie en arbre

Soutenue publiquement le 12 décembre 2011 devant le jury

Rapporteurs :	Mme Pascale Minet	Chercheur HDR à l'INRIA
	M. Thierry Val	Professeur à l'Université de Toulouse
Examinateurs :	M. Tuan Dang	Expert Ingénieur Recherche à EDF
	Mme Isabelle Guérin-Lassous	Professeur à l'Université de Lyon I
Directeurs :	M. Michel Misson	Professeur à l'Université d'Auvergne
	M. Alexandre Guitton	Maître de conférences à l'Université Blaise Pascal

Remerciements

Ce travail a été effectué au laboratoire LIMOS, au sein de l'équipe Réseaux et Protocoles, sous la direction de Monsieur Michel Misson. Je le remercie sincèrement de m'avoir fait l'honneur de m'accueillir dans son équipe et d'avoir été mon directeur de thèse. Je lui suis très redevable pour sa rigueur et pour ses conseils constructifs.

Que Monsieur Alexandre Guitton, mon encadrant de thèse, reçoive toute l'expression de ma reconnaissance pour avoir participé à l'encadrement mon sujet de recherche et pour tout son dynamisme et ses compétences scientifiques qui m'on permis de mener à bien cette étude. Je le remercie pour sa bonne humeur communicative qui fait que travailler avec lui est un plaisir, ainsi que pour sa rigueur et sa clarté qui nous ont permis d'avancer très rapidement sur un sujet à la fois riche et prometteur.

Je tiens à remercier le Conseil National de Recherche Scientifique Libanais (CNRS-L) pour avoir accepté de financer mon séjour en France, et en particulier Monsieur Charles Tabet pour son immense aide et son support infini.

Je remercie tout particulièrement Madame Pascale Minet, HDR et chargée de recherche à l'INRIA Rocquencourt, ainsi que Monsieur Thierry Val, Professeur à l'Université de Toulouse, qui ont accepté de juger ce travail et d'en être les rapporteurs, ainsi que pour leur lecture très approfondie de mon manuscrit qui a permis d'en améliorer grandement la qualité.

Je suis très sensible à la présence dans ce jury de Madame Isabelle Guérin Lassous, Professeur à l'Université de Lyon I, qui a accepté de présider le jury de cette thèse. Je la remercie pour ses

encouragements.

Je tiens également à remercier Monsieur Tuan Dang, docteur et expert ingénieur de recherche à l'EDF, pour avoir accepté de participer au jury de cette thèse.

J'adresse mes remerciements aux membres de l'équipe Réseaux et Protocoles pour leurs encouragements. J'ai vécu avec eux des moments inoubliables. Je remercie en particulier François Delobel, pour toutes les discussions importantes que nous avons eues notamment sur les preuves théoriques de ma thèse, Frédérique Jacquet, que je remercie pour sa bonté et la sympathie qu'elle m'a témoignée et pour son aide à la préparation du pôt. Je remercie Noura Guerchouch pour son aide immense et sa disponibilité pour l'impression de mon manuscrit de thèse.

Je voudrais remercier tous les enseignants avec qui j'ai eu l'opportunité de travailler. C'est avec plaisir que je remercie Guénal Davalan pour sa disponibilité et son support infini durant les TP de réseaux. Je remercie également Joël Toussaint pour son aide et ses encouragements.

Toute mon amitié à Sabri pour son humour et sa bonté, à Nassima pour les hauts et les bas que j'ai partagé avec elle. Je remercie mon pote Malick pour ses efforts qu'il a fait pour parler le libanais et je lui dis « dewekhon Rimone ». Je remercie aussi Lionel pour sa bonté, ses encouragements, et le soutien moral qu'il m'a offert tout au long de cette thèse. Je remercie Fiona pour son amitié et son immense gentillesse. J'adresse mes remerciements à ma chère amie Adeline pour toutes ses ondes positives qu'elle m'a offert (et tout). Je remercie mon amie Maya pour les gâteaux orientaux (et pour « emm tayssir »). Je remercie aussi mes amis libano-clermontois, surtout Azzam, Fadi, Younis et Yasser, pour leur gentillesse et leurs encouragements (et pour « Abou Ali ») et Amine je lui demande « mine lli fatah l raddi ? ». Je tiens à remercier Sahar, qui, bien que je ne la connaisse pas depuis longtemps, est plus qu'une sœur pour moi à Clermont. Je la remercie pour son immense aide dans l'organisation du pôt de ma soutenance. À eux tous, je souhaite beaucoup de bien.

Je tiens à remercier aussi mes amis du Liban. Je voudrais exprimer mon amitié envers mon amie Nancy qui m'a suivie jour par jour (merci pour son bonjour chaque matin). Je remercie mes amis de l'USEK, en particulier, Charles et Roy, pour leurs encouragements, leur bonté et leur foi en moi.

Finalement, je voudrais exprimer tout mon amour et ma reconnaissance à mes proches pour m'avoir aidé et encouragé durant mes longues études, et pour m'avoir toujours soutenu moralement et financièrement. Je pense particulièrement à mon père et à ma mère, à qui je dédie l'ensemble de mon travail, et à toute ma famille, toujours unie dans les épreuves. J'embrasse, au passage, mon petit neveu Giorgio, nouvellement entré dans la famille.

Table des matières

Table des figures

Table des figures

Introduction

Depuis leur invention il y a quelques années, les réseaux de capteurs sans fil ont connu un succès grandissant au sein des communautés scientifiques et industrielles. Grâce aux avantages qu'ils offrent, ces réseaux de capteurs sans fil ont pu s'instaurer comme acteurs incontournables dans les architectures réseaux actuelles. Le médium sans fil offre en effet des propriétés uniques, qui peuvent être résumées en trois points : la facilité du déploiement, l'ubiquité de l'information et le coût réduit de leur installation.

Les réseaux de capteurs sans fil sont très populaires grâce à leur autonomie énergétique et à leur facilité de déploiement. En pratique, le déploiement des capteurs dépend de l'application. Il peut être aléatoire, pour la surveillance de sites naturels étendus par exemple, ou bien précis, pour le suivi d'une activité industrielle par exemple. Actuellement, ce type de réseaux envahit plusieurs domaines d'applications : le domaine de l'écologie [KR04] pour la surveillance des polluants, le domaine médical [BM04] pour la surveillance des patients et pour la collecte d'informations physiologiques de meilleure qualité, le domaine militaire [SM04] pour la surveillance de troupes, ou encore le domaine industriel [AACG+09] pour la surveillance de chaînes de production.

Les objectifs des réseaux de capteurs sans fil sont l'économie d'énergie pour prolonger la durée de vie du réseau, le faible délai de transmission et de récupération de l'information pour garantir un niveau acceptable aux performances du réseau, et l'auto-configuration des entités. À ces besoins se rajoutent la tendance actuelle d'avoir un seul réseau pour supporter plusieurs

applications au lieu d'avoir un réseau différent pour chacune [CHCP07]. Chaque application peut générer plusieurs types de trafic (par exemple, un trafic périodique, un trafic de contrôle, des alarmes, ...) qu'il faut gérer de manière approuvée en fonction des exigences de qualité de service (QoS, pour *Quality of Service*).

Dans cette thèse, nous étudions les techniques de communications inter-couches, nommées *cross-layering*, pour améliorer les performances et fournir de la QoS, en considérant principalement la sous-couche MAC et les protocoles de routage conjointement.

Nous centrons notre étude sur le protocole MaCARI, qui est un protocole d'accès au médium déterministe et économe en énergie, adapté aux réseaux de capteurs sans fil [CLG$^+$09] proposé dans le cadre du projet ANR OCARI (pour Optimisation des Communications Ad hoc pour les Réseaux Industriels). MaCARI adopte une segmentation temporelle qui améliore les performances du réseau et permet de garantir l'accès au médium pour un trafic de type prioritaire. MaCARI garantit de même une borne supérieure connue au délai de transit des informations prioritaires. Les techniques utilisées dans MaCARI réalisent un *cross-layering* entre la sous-couche MAC et la couche réseau que nous allons développer. Notre contribution a pour but d'améliorer les performances du protocole MaCARI vis à vis du débit offert et du temps de transit associé aux trames.

À mesure que de nouvelles applications continuent d'émerger, il est probable que des réseaux de capteurs intégrant de nombreuses QoS vont devenir nécessaires pour faire face au nombre croissant de types de trafic et de QoS. Nous généralisons donc le concept de MaCARI en proposant une architecture multi-couches où n combinaisons de protocoles MAC-routage $(\mathcal{M}_i, \mathcal{R}_i)$ sont utilisées. Toutes les entités (c'est-à-dire tous les nœuds) sont synchronisées et une file d'attente $\mathcal{Q}_i$ est associé à chaque couple $(\mathcal{M}_i, \mathcal{R}_i)$. À un instant t donné, tous les nœuds fonctionnent selon une seule combinaison. Chaque combinaison est activée pour un intervalle de temps précis. Le but de notre architecture est de pouvoir profiter de ces combinaisons afin d'offrir alternativement différentes QoS. Cependant, cette architecture cause un problème de dimensionnement des intervalles de temps alloués aux combinaisons $(\mathcal{M}_i, \mathcal{R}_i)$ ce qui a un impact sur le débit et la latence du réseau. Pour résoudre ce problème, nous proposons d'appliquer le *cross-layering* en échangeant les paquets entre les files d'attente des différentes combinaisons quand c'est possible. Durant l'intervalle de temps qui lui est associée, chaque combinaison $(\mathcal{M}_i, \mathcal{R}_i)$ traite tous les paquets de sa file d'attente $\mathcal{Q}_i$, ainsi que les paquets de la file d'attente d'autres périodes si la durée restante le permet. Ceci constitue une hypothèse de base de notre contribution.

Ce travail a été réalisé au sein de l'équipe Réseaux et Protocoles du Laboratoire d'Informa-

tique, de Modélisation et d'Optimisation des Systèmes (LIMOS) de l'Université Blaise Pascal, à Clermont-Ferrand, sous la direction de Michel Misson et l'encadrement d'Alexandre Guitton.

Ce mémoire est organisé comme suit.

Dans le chapitre 1, nous présentons un état de l'art des travaux existants sur lesquels nous nous basons pour proposer nos contributions. Nous étudions tout d'abord les principaux mécanismes des standards IEEE 802.15.4 et ZigBee, qui constituent une pile protocolaire qui a été conçue spécialement pour les réseaux de capteurs. La sous-couche MAC du standard IEEE 802.15.4 propose une période d'inactivité pour économiser l'énergie et la couche réseau de ZigBee propose un protocole de routage qui ne nécessite pas d'échanges périodiques de messages de contrôle supplémentaires (ou de tables de routage). Ensuite, nous dressons un état de l'art sur le dimensionnement temporel de l'activité des nœuds en étudiant les dimensionnements statiques et dynamiques. Enfin, nous détaillons les différentes techniques de *cross-layering* ainsi que leur utilité dans les réseaux de capteurs, en nous concentrant sur le *cross-layering* entre la sous-couche MAC et la couche réseau.

Dans le chapitre 2, qui est la première contribution de cette thèse, nous proposons une architecture multi-couches inspirée de MaCARI, permettant de combiner plusieurs protocoles MAC-routage en découpant le temps en périodes et en activant une combinaison de protocoles à chaque période. Cette architecture permet de garantir plusieurs QoS dans un même réseau et peut supporter plusieurs applications. En effet, chaque combinaison de protocoles MAC-routage permet d'acheminer un type de trafic d'une façon efficace. L'architecture multi-couches cause un problème de dimensionnement des intervalles de temps réservés pour chaque combinaison : un mauvais dimensionnement influe négativement sur le débit et la latence du réseau. En effet, le temps risque d'être gaspillé si la quantité de trafic est faible par rapport à la capacité offerte par l'intervalle de temps, et le trafic risque d'être perdu si la période est trop petite pour supporter la charge du trafic. Pour résoudre le problème de dimensionnement, nous proposons de permettre les échanges entre les files d'attente des nœuds. Nous montrons que l'échange ne peut pas être fait pour n'importe quelle combinaison parce que des détours sont susceptibles d'apparaître dans le réseau à cause du routage.

Dans le chapitre 3, nous intégrons notre mécanisme d'échanges de files d'attente au sein de la méthode d'accès MaCARI. Dans la version étudiée ici, MaCARI dispose de deux périodes d'activité : une période ordonnancée avec un mécanisme de type TDMA et le protocole de routage hiérarchique de ZigBee, et une période non ordonnancée avec le mécanisme CSMA/CA slotté et une version optimisée du protocole de routage hiérarchique. Nous discutons de l'échange

des files d'attente dans MaCARI avec un seul type de trafic, puis avec deux types de trafic.

Dans le chapitre 4, nous présentons la démarche de simulation des mécanismes proposés dans les chapitres précédents. Nous montrons l'utilité de l'architecture multi-couches, nous identifions le risque d'apparition de détours (ou de boucles) dans le réseau et nous montrons comment l'éviter. Nous testons ensuite l'échange des files d'attente dans MaCARI avec un seul type de trafic, puis avec deux types de trafic. Nous montrons le gain de performances du réseau en utilisant notre contribution par rapport aux performances de base fournies par MaCARI.

Dans le chapitre 5, nous proposons des contributions complémentaires permettant de réduire la congestion dans un réseau. Nous proposons des solutions de déploiement qui visent à réduire le temps d'installation d'un réseau permettant d'activer les nœuds d'une manière séquentielle. Nous proposons ensuite un protocole de routage pour la transmission d'un trafic prioritaire (de type alarmes). Ce protocole est basé sur des pivots et consiste à réduire la congestion dans le réseau sauf au niveau du puits. Pour réduire la congestion au niveau du puits, nous proposons une amélioration de ce protocole qui considère plusieurs puits conjointement.

Nous concluons ce mémoire en résumant nos contributions et en discutant des différentes perspectives qui découlent de ce travail.

Chapitre 1

État de l'art

Les réseaux de capteurs sans fil sont de plus en plus utilisés pour surveiller l'environnement et détecter des événements critiques. Ces réseaux sont composés de plusieurs entités de faible capacité, alimentées par des petites batteries. Le but de ces réseaux est généralement d'envoyer à une entité centrale, de capacité plus grande que celle des autres entités du réseau, des informations précises dépendant des applications supportées par le réseau. Ces informations doivent être transmises au plus tôt en tenant compte de la contrainte de consommation d'énergie.

Les standards IEEE 802.15.4 et ZigBee ont été proposés pour l'implémentation des couches basses et hautes (respectivement) de la pile protocolaire des réseaux à faible débit et à faible consommation d'énergie. Ils permettent de déployer un réseau de capteurs répondant à certains critères de délais et de taux de pertes, et respectant des contraintes de consommation d'énergie. Ces standards découpent le temps en périodes. Le dimensionnement des périodes est lié à l'application et à la génération du trafic d'une part, et à la taille du réseau et à la politique choisie pour l'accès au médium d'autre part.

1.1 Introduction

Les réseaux de capteurs sans fil disposent de ressources limitées en termes de capacité de calcul, d'espace mémoire et de ressources énergétiques. Ces contraintes sont souvent ignorées par les protocoles MAC traditionnels qui essaient d'augmenter le débit et de minimiser le délai de transit tout en restant robustes par rapport aux conditions du réseau.

Le défi est de pouvoir réduire les risques de transmissions infructueuses (dues à des collisions

par exemple) dans un réseau de capteurs sans fil tout en respectant ses contraintes (par exemple, économie d'énergie) et les besoins des applications. La nature des applications requiert non seulement la prise en compte de l'économie d'énergie, mais aussi le respect d'une qualité de service.

Ce chapitre présente un état de l'art des travaux existants sur lesquels nous nous basons dans la suite de cette thèse afin d'améliorer les performances de la pile protocolaire OCARI. Rappelons que notre but est d'améliorer les performances des réseaux de capteurs en termes de délai, de débit et de taux de perte ; ceci en autorisant une communication inter-couches *(cross-layering)* entre la sous-couche MAC et la couche Réseau. Cette solution doit tenir compte de la contrainte énergétique des nœuds et maintenir de performances requises pour les applications du réseau.

Par la suite, nous détaillons les standards IEEE 802.15.4 et ZigBee qui sont incontournables dans le domaine des réseaux de capteurs sans fil. Ensuite, nous décrivons le dimensionnement temporel pour les réseaux de capteurs sans fil. Puis, nous présentons quelques travaux sur la technique du *cross-layering* dans un but d'optimiser les performances du réseau.

1.2 Standard IEEE 802.15.4

Le standard IEEE 802.15.4 [IEE06] est le standard le plus utilisé dans les réseaux de capteurs sans fil. Il permet d'interconnecter des capteurs, des actionneurs et des unités de traitement qui constituent l'infrastructure nécessaire pour surveiller un environnement physique [ZL04, CGH+02]. Dans ce qui suit, nous décrivons le standard IEEE 802.51.4 de base sachant que des versions plus récentes, comme IEEE 802.15.4a [IEE07], existent.

1.2.1 Description

Le standard IEEE 802.15.4 définit la couche physique (PHY) et la sous-couche d'accès au médium (MAC) de la pile protocolaire d'un réseau personnel sans fil à faible débit (LR-WPAN, pour *Low Rate Wireless Personal Area Network*). La couche PHY fonctionne sur plusieurs bandes de fréquences, incluant la bande ISM (pour Industriel, Scientifique, et Médical) à 2,4 GHz. La sous-couche MAC utilise le mécanisme CSMA/CA (pour *Carrier Sense Multiple Access with Collision Avoidance*) pour accéder au médium. Deux modes de fonctionnement peuvent être utilisés : le mode avec suivi de balises *(beacon-enabled mode)* utilisant le CSMA/CA slotté, et le mode sans suivi de balises *(non beacon-enabled mode)* utilisant le CSMA/CA non

slotté.

Le standard IEEE 802.15.4 supporte deux types de topologies : la topologie en étoile et la topologie pair-à-pair. La topologie en étoile est composée d'un nœud central, nommé coordinateur du PAN (pour *Personal Area Network*), et de nœuds feuilles. Les feuilles communiquent entre elles en passant systématiquement par le coordinateur du PAN. Le coordinateur du PAN gère les associations des feuilles au réseau. La taille du réseau dans la topologie en étoile est donc limitée à la portée des entités impliquées. La topologie pair-à-pair est composée de trois types de nœuds : le coordinateur du PAN, des coordinateurs et des feuilles. Chaque entité (sauf la feuille) peut communiquer avec les nœuds qui sont à sa portée. Le coordinateur du PAN a un ou plusieurs coordinateurs voisins qui peuvent eux aussi gérer les associations d'autres coordinateurs et feuilles. Le réseau est alors plus complexe à gérer mais son envergure est plus étendue.

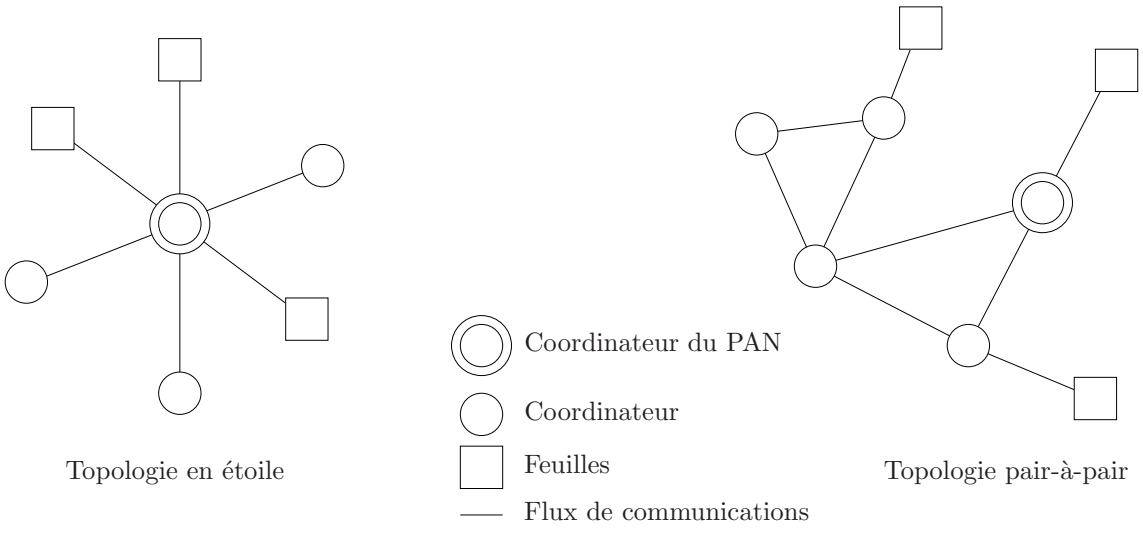


Figure 1.1 – Types de topologies disponibles pour le standard IEEE 802.15.4.

La figure 1.1 illustre deux exemples de topologies : une topologie en étoile et une topologie pair-à-pair. Le coordinateur du PAN est représenté par un double cercle, les coordinateurs par des cercles et les feuilles par des carrés. Les lignes représentent les communications possibles entre les entités.

Une entité IEEE 802.15.4 peut être un FFD (pour *Full Function Device*) ou bien un RFD (pour *Reduced Function Device*) selon ses capacités et ses ressources disponibles. Le FFD dispose de toutes les fonctionnalités disponibles dans le standard. Il peut fonctionner en tant que coordinateur du PAN, coordinateur ou bien feuille. Le RFD, en revanche, est une entité allégée en fonctionnalité. Il ne peut avoir que le rôle d'une feuille. Un FFD peut communiquer avec

d'autres FFD et d'autres RFD, alors qu'un RFD ne peut communiquer qu'avec un seul FFD.

Dans ce qui suit, nous détaillons la couche PHY et la sous-couche MAC du standard IEEE 802.15.4.

1.2.2 Couche physique

La couche physique du standard IEEE 802.15.4 initial fournit les services de gestion du module radio et de transmission de bits sur le médium radio. Elle réalise la sélection des canaux, la modulation du signal et gère la puissance d'émission. Les informations (découpées en trames de tailles variables par la sous-couche MAC) sont envoyées selon une modulation donnée. Le débit proposé varie en fonction de la fréquence et de la modulation [Bur01] comme le montre le tableau 1.1. L'utilisation des fréquences dépend de l'environnement et de la législation (qui est variable selon les pays). Cette couche est généralement intégrée dans le module radio.

Bande de fréquences (MHz)	**Modulation**	**Débit (kbps)**	**Nombre de canaux**
$868 - 868,6$	BPSK	20	1
$868 - 868,6$	ASK	250	
$868 - 868,6$	O-QPSK	100	
$902 - 928$	BPSK	40	10
$902 - 928$	ASK	250	
$902 - 928$	O-QPSK	250	
$2400 - 2483,5$	O-QPSK	250	16

Tableau 1.1 – Débit selon la modulation et la fréquence.

Les fréquences basses permettent généralement d'atteindre de plus grandes portées (due à de plus faibles pertes lors de la propagation). Les fréquences plus grandes permettent d'obtenir une latence faible (car le débit est plus grand), un nombre de canaux plus important (à taille de bandes de fréquences égales), mais conduisent à une portée réduite. La couche physique du standard IEEE 802.15.4 offre au total 27 canaux sur l'ensemble des trois bandes de fréquences. Elle prend en charge les tâches suivantes :

- L'activation et la désactivation du module radio : à un moment donné, le module radio peut être soit en mode de transmission, soit en mode de réception, soit dans un des modes de sommeil.
- La détection de l'énergie sur le canal courant : il s'agit d'une estimation de l'énergie reçue sur un canal. Cette estimation est utilisée pour la sélection du canal et pour l'échantillonnage du canal (CCA, pour *Clear Channel Assessment*), afin de déterminer si le canal est libre ou s'il est occupé.

– L'indication de la qualité du lien : cette indication évalue la qualité du signal reçu suite à la réception correcte d'une trame sur un lien.
– La sélection du canal : la couche physique doit être capable de faire fonctionner son module radio sur le canal spécifié par les couches supérieures. En effet, une seule couche PHY ne peut écouter ou transmettre que sur un canal à la fois.

1.2.3 Sous-couche MAC

La sous-couche MAC du standard IEEE 802.15.4 gère l'accès au médium. Elle contrôle les opérations liées à la création du réseau et aux associations entre les entités du réseau, les échanges de trames ainsi que la synchronisation par envoi de balises *(beacon)* (dans le mode avec suivi de balises). Une balise est une trame diffusée portant des informations de service et de synchronisation principalement.

En mode sans suivi de balises, il n'y a pas de synchronisation. Les balises ne servent qu'à l'association au réseau. Les coordinateurs sont toujours actifs alors que les feuilles peuvent n'être actives que seulement lorsqu'elles souhaitent communiquer. L'accès au médium n'est pas garanti : les entités sont toujours en concurrence pour émettre.

En mode avec suivi de balises, chaque coordinateur envoie périodiquement une balise pour synchroniser l'activité des entités et les associer au réseau. La balise contient des informations décrivant la structure d'un cycle. Ce cycle, nommé supertrame (pour *superframe*), est découpé en deux périodes principales : une période d'activité suivie d'une période d'inactivité.

Structure de la supertrame dans le mode avec suivi de balises

Une supertrame est délimitée par la transmission (ou la réception) d'une balise, et possède une période d'activité et une période d'inactivité (éventuellement vide). Toutes les entités sous la responsabilité d'un coordinateur sont synchronisées grâce à la balise reçue et partagent la même période d'activité. Toutes les stations peuvent passer en mode sommeil dès qu'elles se trouvent dans la période d'inactivité.

La période d'activité est composée de deux parties : la période de contention, appelée CAP (pour *Contention Access Period*), et la période sans contention, appelée CFP (pour *Contention Free Period*). La balise est envoyée par le coordinateur au début de la CAP. La CAP définit un intervalle de temps pendant lequel tous les nœuds entrent en concurrence pour accéder au médium (avec le mécanisme de CSMA/CA slotté (pour *slotted Carrier Sense Multiple Access with Collision Avoidance*). La CAP est découpée en seize intervalles de temps de durées égales.

La CFP est optionnelle. Quand elle existe, elle permet de garantir l'accès au médium par le biais d'intervalles de temps attribués à un couple feuille-coordinateur et appelés GTS (pour *Guaranteed Time Slots*). L'allocation des GTS se fait à la demande des nœuds durant la CAP. Un nœud peut réserver des GTS dont la durée est comprise entre un et sept intervalles de temps. Le coordinateur peut allouer plusieurs GTS si le cumul de leur durée n'excède pas sept intervalles de temps. Durant la période d'inactivité, les entités peuvent passer en mode sommeil mais doivent se réveiller avant la prochaine transmission ou réception de balise.

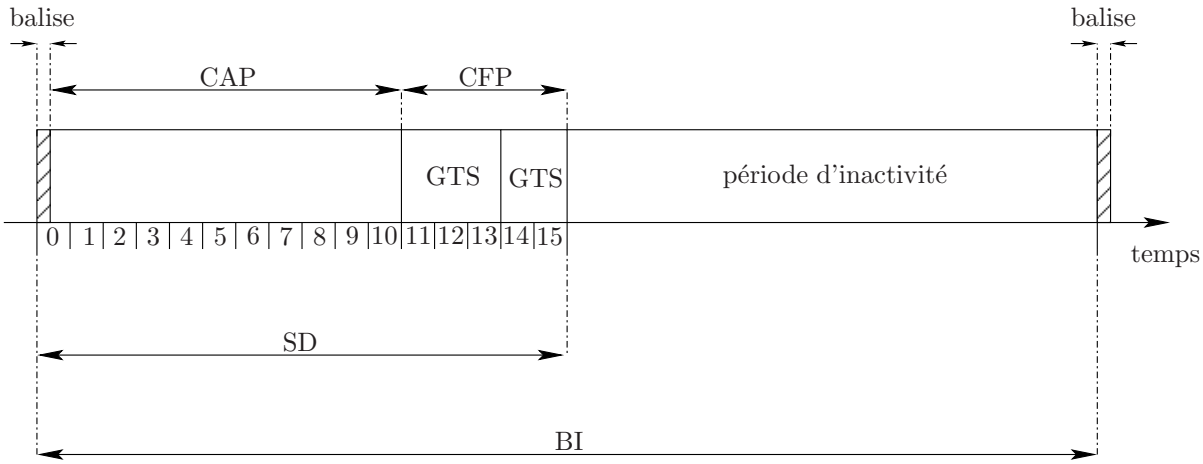


Figure 1.2 – Représentation des différentes périodes de la supertrame.

La figure 1.2 illustre les différentes périodes de la supertrame. La durée entre deux balises, dénotée par BI (pour *Beacon Interval*) sur la figure, est déterminée par le paramètre BO (pour *macBeaconOrder*). La durée de la période d'activité, dénotée SD (pour *Superframe Duration*) est déterminée, quant-à elle, par le paramètre SO (pour *macSuperframeOrder*). Les valeurs de ces deux périodes sont données par les formules suivantes :

$$\begin{cases} \text{BI} = aBaseSuperframeDuration \cdot 2^{\text{BO}}, \\ \text{SD} = aBaseSuperframeDuration \cdot 2^{\text{SO}}, \end{cases}$$

avec $0 \leq \text{SO} \leq \text{BO} \leq 14$. *aBaseSuperframeDuration* est un attribut de la sous-couche MAC qui est égal par défaut à $15,36$ ms. Notons que quand SO est égal à BO, la supertrame ne dispose pas d'une période d'inactivité et les nœuds sont toujours actifs.

1.2.4 Le mécanisme de CSMA/CA

La sous-couche MAC du standard IEEE 802.15.4 dispose de plusieurs façons d'accéder au médium radio, notamment selon le type de trames à envoyer. Les balises et les acquittements

sont toujours envoyés de façon préemptive à des instants décidés par le nœud émetteur. En mode avec suivi de balises, durant la CAP, l'accès au médium est géré selon le mécanisme de CSMA/CA slotté, et durant la CFP, les trames d'un couple feuille-coordinateur sont envoyées directement pendant les GTS correspondants. En mode sans suivi de balises, les trames sont envoyées en utilisant le mécanisme de CSMA/CA non slotté.

CSMA/CA slotté

Le CSMA/CA slotté est une variante de CSMA/CA à la base de IEEE 802.11. Son mécanisme est cadencé par une unité de temps appelée période de *backoff* et égale au temps d'envoi effectif de 20 symboles (équivalent à 10 octets) par le module radio, soit 320 μs. Les entités sont synchronisées sur cette échelle de temps grâce à la réception de la balise. Les *backoffs* servent à discrétiser le temps afin de réduire le temps de surveillance du médium et de garantir une économie d'énergie, point critique dans les réseaux de capteurs sans fil. Ce mécanisme implique trois paramètres :

- Le nombre de tentatives d'accès au canal, NB (pour *Number of Backoffs*), est le nombre de fois où le mécanisme de tirage de *backoff* est appliqué pour une trame. Cette valeur doit être initialisée à zéro avant chaque nouvelle transmission. Si NB dépasse la constante *macMaxCSMABackoffs* (qui est, par défaut, fixée à 4 dans le standard), une notification d'échec est retournée au niveau supérieur.
- La fenêtre d'écoute, CW (pour *Contention Window*), définit le nombre de périodes de *backoff* pendant lesquels le canal doit rester libre avant que la transmission ne commence. La valeur de CW est initialisée à deux avant chaque tentative de transmission, et réinitialisée à deux à chaque fois que le canal est détecté occupé.
- L'exposant de la fenêtre du *backoff*, BE (pour *Backoff Exponent*), permet de calculer la taille de la fenêtre dans laquelle est tiré le nombre de périodes de *backoff* qu'un nœud doit attendre avant de tester si le canal est libre. Plus BE est petit, plus la fenêtre $[0; 2^{(\mathrm{BE})} - 1]$ est petite.

La figure 1.3 représente les différentes étapes du mécanisme CSMA/CA slotté. Les cinq étapes sont les suivantes :

- L'étape 1 correspond à l'initialisation des paramètres. Quand le mécanisme CSMA/CA slotté est utilisé, la sous-couche MAC initialise tout d'abord les paramètres NB, CW et BE. Ensuite, elle vérifie si le paramètre de vie étendue *(extended life)* est positionné. Si c'est le cas, la valeur de BE est initialisée au minimum entre 2 et *macMinBE*. Sinon, BE est

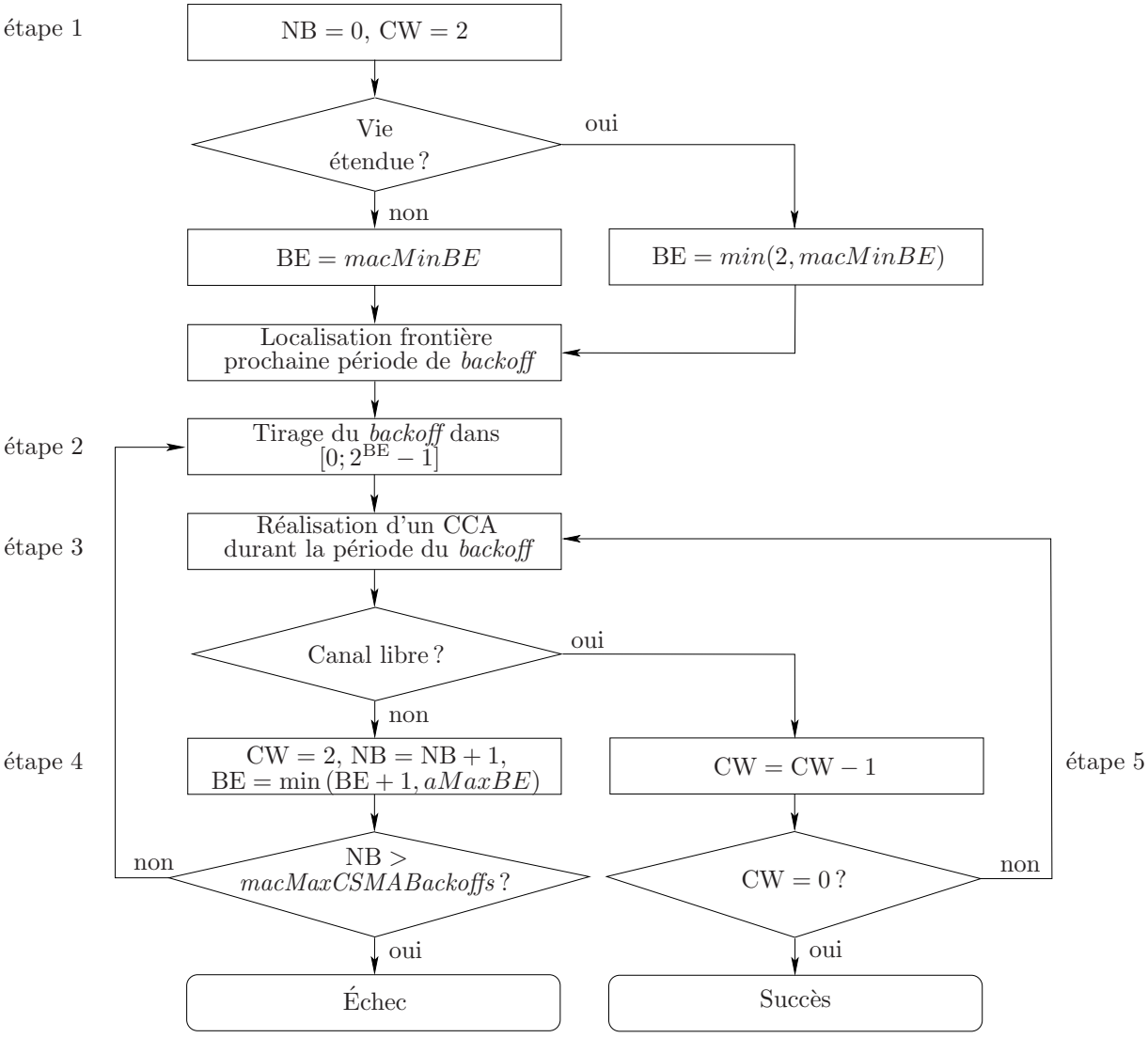


Figure 1.3 – Mécanisme CSMA/CA slotté.

initialisé avec la valeur *macMinBE*. *macMinBE* vaut 3 par défaut. Ensuite la sous-couche MAC localise le début de la prochaine période de *backoff*.

– L'étape 2 correspond à l'attente. Le nœud attend pendant un temps appelé *backoff*, correspondant à un nombre tiré aléatoirement dans la fenêtre $[0; 2^{\mathrm{BE}} - 1]$. Si le nombre de périodes de *backoff* tiré est plus grand que le nombre de périodes de *backoff* restant dans la CAP, le mécanisme consomme le *backoff* jusqu'à la fin de la CAP et reporte ce qui lui reste à la CAP de la prochaine supertrame. Quand le nombre de périodes de *backoff* tiré expire, la sous-couche MAC vérifie si le nombre restant de périodes de *backoff* dans la CAP est suffisant pour effectuer CW fois un test de canal, transmettre la trame et recevoir l'acquittement. Si c'est le cas, le mécanisme passe à l'étape suivante. Sinon, la sous-couche MAC reporte le test du canal au début de la CAP de la prochaine supertrame (en effectuant un *backoff* supplémentaire pour éviter les collisions).
– L'étape 3 correspond au test du canal. La sous-couche MAC sollicite la couche physique pour l'échantillonnage du canal. Si le canal est détecté occupé, le mécanisme passe à l'étape 4. Sinon, il passe à l'étape 5.
– L'étape 4 correspond à la détection du canal occupé. Si le canal est occupé, le paramètre CW est réinitialisé à deux et les paramètres BE et NB sont incrémentés. BE ne peut pas dépasser la constante *aMaxBE* fixée à cinq. Si NB est inférieur à *macMaxCSMABackoffs* (qui vaut 4 par défaut), le mécanisme retourne à l'étape 2. Sinon, il y a un échec d'accès au canal.
– L'étape 5 correspond à la détection du canal libre. Si le canal est libre, le paramètre CW est décrémenté. Quand CW devient égal à zéro, la sous-couche MAC envoie la trame. Sinon, le mécanisme retourne à l'étape 3.

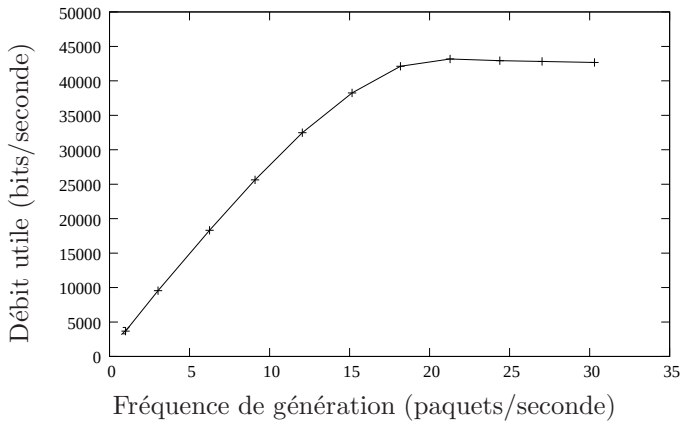


Figure 1.4 – Débit utile en fonction de la fréquence de génération de paquets

La figure 1.4 est extraite de la figure 3.22 de [Had11]. Les résultats illustrés dans cette figure sont obtenus en utilisant le module `wpan` de NS-2 [NS202]. La topologie du réseau est une topologie en étoile, composée d'un coordinateur du PAN et de 10 nœuds. Les nœuds sont déployés de sorte que chacun est à la portée de tous les autres nœuds pour éviter l'effet du terminal caché. La valeur de BO est fixée à 6 et celle de SO à 5. La figure montre le débit utile en fonction de la fréquence de génération de paquets pour l'algorithme CSMA/CA slotté. Le débit utile augmente avec la fréquence de génération de paquets et atteint un état stable à partir de la fréquence de génération de 21 paquets par seconde. Dans le cas de non saturation, le débit utile augmente avec la fréquence de génération des paquets ce qui est expliqué par le fait que le nombre de trames envoyées avec succès augmente. Dans le cas de saturation, le nombre de trames envoyées avec succès reste constant. En effet, en cas d'engorgement du réseau, le CSMA/CA slotté du standard IEEE 802.15.4 purge les trames très rapidement (le nombre de tirage de *backoffs* pour une même trame, *macMaxCSMABackoffs*, vaut 4 par défaut, et le nombre maximum de tentatives d'accès au médium, *macMaxFrameRetries*+1, vaut $3 + 1 = 4$).

CSMA/CA non slotté

Le mécanisme du CSMA/CA non slotté est utilisé dans le mode sans suivi de balises. Il utilise aussi les périodes de *backoff* comme référence de temps. Contrairement au CSMA/CA slotté, les nœuds ne sont pas synchronisés pour accéder au médium. La transmission commence dès que le médium est détecté libre, et l'envoi n'est pas synchronisé sur les débuts d'une période de *backoff*. De plus, un seul test de canal est fait pour déterminer s'il est libre ou occupé (ce qui correspond à $\mathrm{CW} = 1$).

La figure 1.5 représente les différentes étapes du mécanisme CSMA/CA non slotté.

- L'étape 1 correspond à l'initialisation des paramètres. La sous-couche MAC initialise le paramètre NB à zéro et le paramètre BE à *macMinBE*.
- L'étape 2 correspond à l'attente. Le nœud attend pendant un temps appelé *backoff*, correspondant à un nombre tiré aléatoirement dans la fenêtre $[0; 2^{\mathrm{BE}} - 1]$. Quand ce temps expire, le nœud teste le canal.
- L'étape 3 correspond au test du canal. Le nœud teste le canal. Si le canal est détecté occupé, le mécanisme passe directement à l'étape 4. Sinon, il passe à l'étape 5.
- L'étape 4 correspond à la détection du canal occupé. Si le canal n'est pas libre, les paramètres NB et BE sont incrémentés. BE ne peut pas dépasser la constante *aMaxBE*. Si NB est inférieur ou égal à *macMaxCSMACABackoffs*, le mécanisme retourne à l'étape 2.

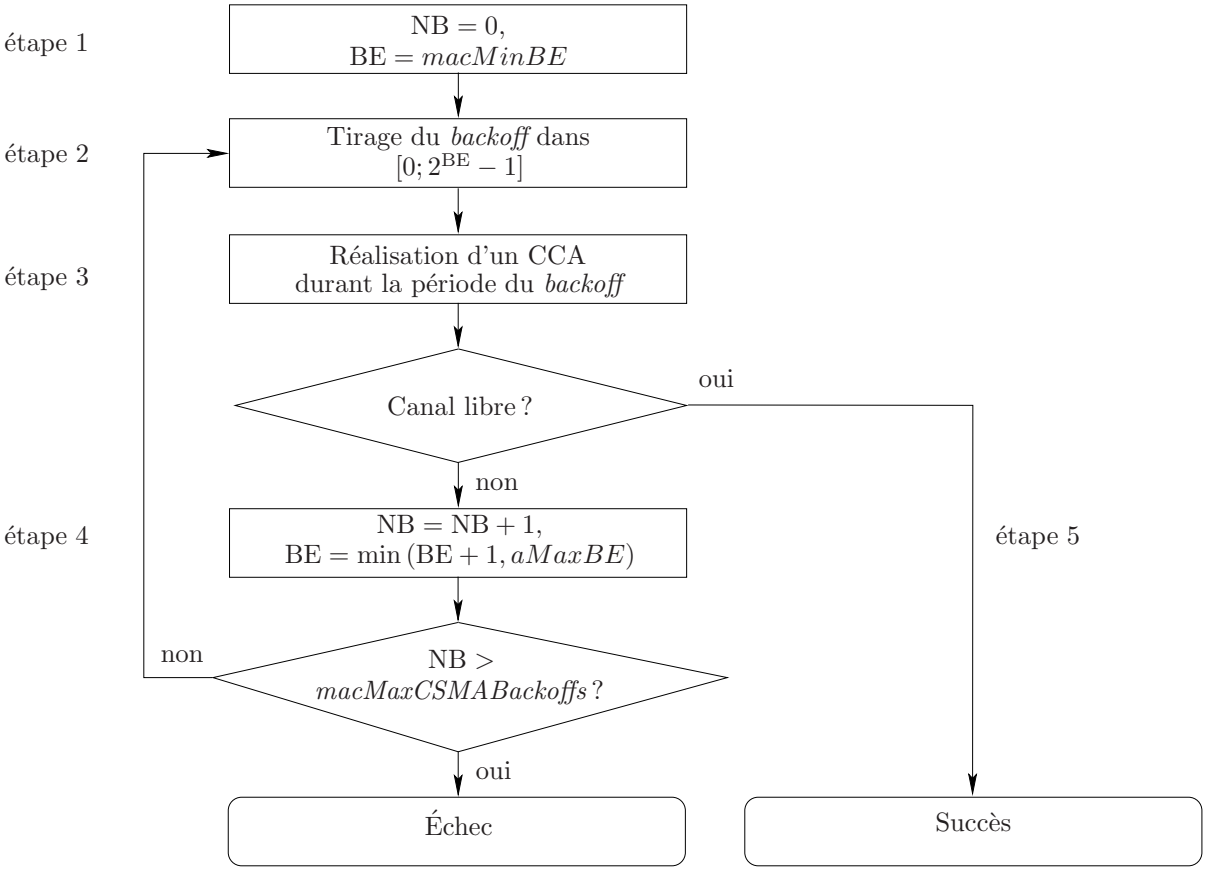


Figure 1.5 – Mécanisme CSMA/CA non slotté.

Sinon, il y a un échec d'accès au canal.

– L'étape 5 est la détection du canal libre. Si le canal est libre, la sous-couche MAC démarre la transmission d'une trame.

1.3 Standard ZigBee

Le standard ZigBee [Zig08] est défini et soutenu par la ZigBee Alliance [All]. Il spécifie les couches supérieures d'un LR-WPAN, en s'appuyant sur IEEE 802.15.4 pour les couches basses. Nous nous intéressons principalement dans cette partie à la couche réseau.

La figure 1.6 permet de schématiser la structure de la pile IEEE 802.15.4/ZigBee ainsi constituée. Les couches sont reliées entre elles par des interfaces. Ces interfaces sont :

– NLDE (pour *Network Layer Data Entity*) pour les échanges de primitives de données entre la couche réseau et la couche application,
– NLME (pour *Network Layer Management Entity*) pour les échanges de primitives de contrôle entre la couche réseau et la couche application,
– MLDE (pour *MAC Layer Data Entity*) pour les échanges de primitives de données entre la sous-couche MAC et la couche réseau,
– MLME (pour *MAC Layer Management Entity*) pour les échanges de primitives de contrôle entre la sous-couche MAC et la couche réseau,
– PD (pour *PHY Data Layer*) pour les échanges de primitives de données entre la couche PHY et la sous-couche MAC,
– PLME (pour *PHY Layer Management Entity*) pour les échanges de primitives de contrôle entre la couche PHY et la sous-couche MAC.

1.3.1 Couche réseau

La couche réseau de ZigBee fournit d'une part des fonctionnalités liées à la transmission de données, comme l'encapsulation des données applicatives et le calcul du prochain saut pour le routage des paquets, et d'autre part des fonctionnalités liées à la gestion des tables de routage, à la création de la topologie et à l'allocation des adresses logiques.

La couche réseau du standard ZigBee prévoit l'utilisation de trois types de nœuds pour constituer un réseau. Une entité ZigBee peut donc être :

– Un coordinateur ZigBee (ZC, pour *ZigBee Coordinator*). Cette entité est un FFD et correspond au coordinateur du PAN défini dans le standard IEEE 802.15.4.

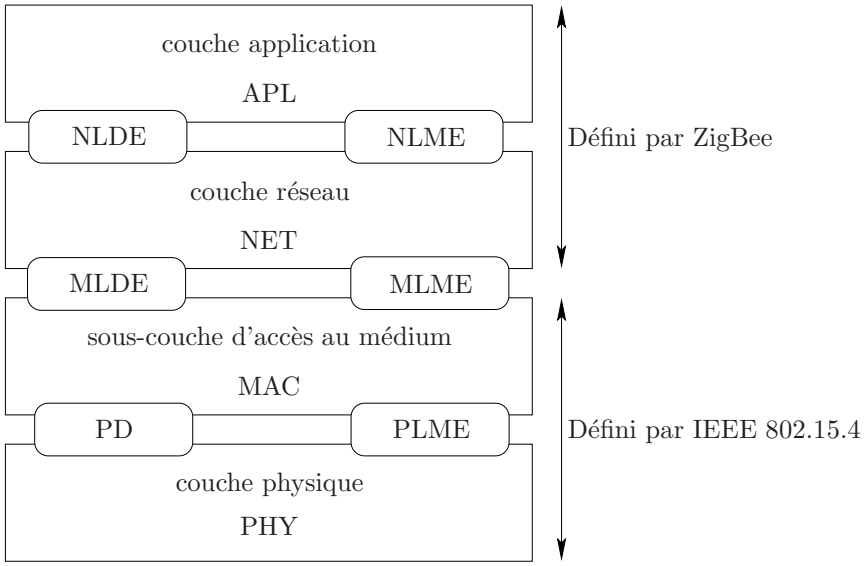


Figure 1.6 – Représentation de la pile IEEE 802.15.4/ZigBee.

- Un routeur ZigBee (ZR, pour *ZigBee Router*). Cette entité est un FFD et correspond au coordinateur défini dans le standard IEEE 802.15.4.
- Un équipement terminal ZigBee (ZED, pour *ZigBee End-Device*). Cette entité est un RFD et correspond à la feuille définie dans le standard IEEE 802.15.4.

Dans la suite, pour des raisons de simplicité, nous utilisons la dénomination du standard IEEE 802.15.4 pour désigner les entités du réseau.

Le standard ZigBee définit trois topologies : la topologie en étoile, la topologie en arbre et la topologie maillée. Ces trois topologies sont illustrées sur la figure 1.7. Dans la topologie en étoile, le réseau est créé et contrôlé par le coordinateur du PAN. Cette topologie est similaire à celle définie dans le standard IEEE 802.15.4. Dans la topologie en arbre, la création de l'arbre est initiée par le coordinateur du PAN. Les coordinateurs et les feuilles souhaitant joindre le réseau sont associés à des coordinateurs qui sont déjà associés à l'arbre du réseau, en formant des liens père-fils. Dans la topologie en arbre, les paquets peuvent être routés selon le routage hiérarchique de l'arbre sans échanges d'informations de routage particulières. Dans la topologie maillée, aucune hiérarchie n'existe entre les coordinateurs, les paquets sont routés selon le protocole de routage AODV (pour *Ad hoc On-demand Distance Vector*) [RFC 3561].

Dans une topologie en arbre, un mécanisme d'allocation d'adresses logiques permet à chaque nœud de gérer les adresses de sa descendance à partir de trois paramètres. Ces trois paramètres sont :

- L_m, qui définit la profondeur maximale du réseau,

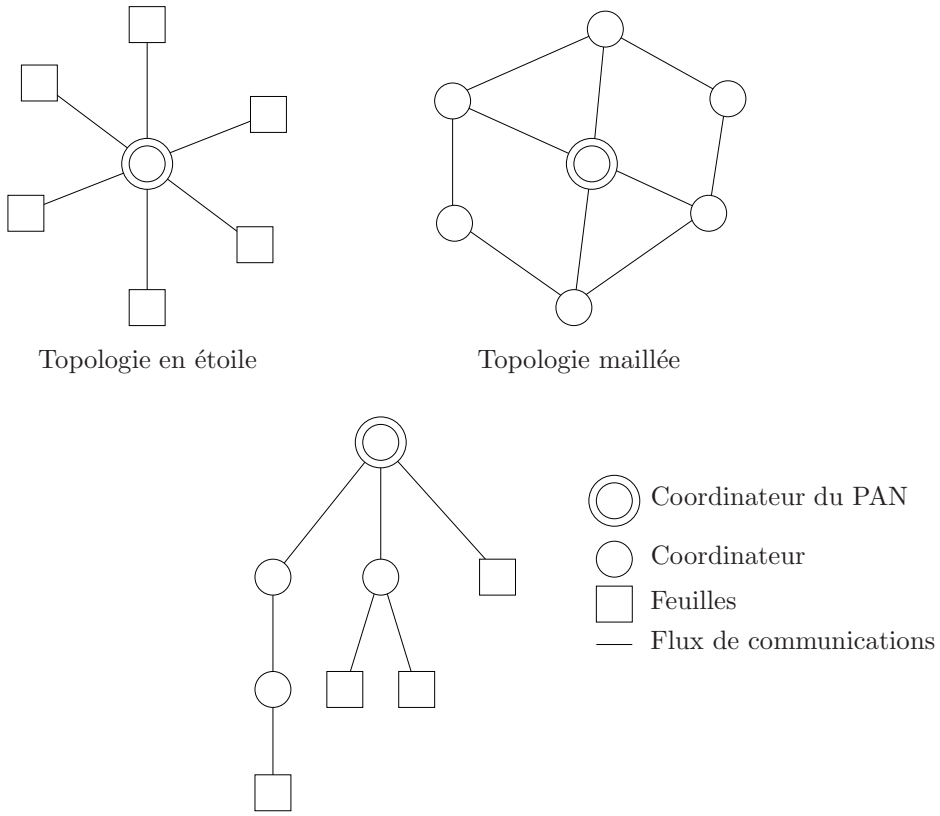


Figure 1.7 – Types de topologies disponibles dans le standard ZigBee.

– C_m, qui définit le nombre maximal de fils par coordinateur,

– R_m, qui définit le nombre maximal de fils coordinateurs par coordinateur.

Le coordinateur du PAN transmet dans la balise ces trois paramètres qui permettent de définir la topologie du réseau.

1.3.2 Allocation des adresses et routage hiérarchique

Chaque entité possède une adresse MAC unique, appelée adresse longue. Cette adresse occupe 8 octets. Le standard ZigBee définit pour chaque entité une adresse logique, dite adresse courte, de 2 octets seulement. Les protocoles utilisent ces adresses afin de réduire la taille des champs d'adressage dans les paquets échangés.

Le standard ZigBee définit deux mécanismes d'allocation d'adresses. Le mécanisme d'allocation d'adresses distribué utilise la structure hiérarchique de l'arbre avec ces trois paramètres : L_m, R_m, et C_m pour allouer les adresses aux entités du réseau. Dans ce cas, le routage est simplifié puisqu'une entité est capable de calculer localement le chemin complet sur l'arbre pour toute destination, en se basant sur l'adresse courte de la destination. Le mécanisme d'allocation d'adresses stochastique alloue les adresses aléatoirement indépendamment des paramètres L_m, R_m, et C_m. Dans ce cas, un protocole de routage sophistiqué comme AODV doit être utilisé.

Dans ce qui suit, nous concentrons notre étude sur le mécanisme d'allocation d'adresses distribué qui évite tout conflit d'adresses [ERGBM10].

Le mécanisme d'allocation d'adresses distribué, appelé DAAM (pour *Distributed Address Allocation Mechanism*), est utilisé pour allouer une adresse unique pour chaque entité du réseau. Ce mécanisme garantit qu'il n'est pas possible que deux nœuds obtiennent la même adresse.

Dans DAAM, le coordinateur du PAN, qui est la racine de l'arbre, possède toujours l'adresse 0. Pour calculer l'adresse d'un fils, chaque nœud du réseau doit connaître les valeurs des trois paramètres : C_m, R_m et L_m. Chaque coordinateur se voit attribuer une plage d'adresses par son père. La première adresse de la plage d'adresses est utilisée comme adresse du coordinateur lui-même, $C_m - R_m$ sont attribués aux éventuelles feuilles et le reste des adresses est distribué entre les R_m coordinateurs fils potentiels. La largeur de la plage d'adresses donnée par un coordinateur C à chaque coordinateur fils, quand C est situé à une profondeur d, est notée par $Cskip(d)$. $Cskip(d)$ est calculé comme suit (cf. paragraphe 3.6.1.6 de [Zig08]) :

$$Cskip(d) = \begin{cases} 1 + C_m \cdot (L_m - d - 1) & \text{si } R_m = 1, \\ \frac{1 + C_m - R_m - C_m \cdot R_m^{L_m - d - 1}}{1 - R_m} & \text{sinon.} \end{cases}$$

Le coordinateur du PAN est en charge de la plage d'adresses globale. Un coordinateur qui possède un $Cskip(d)$ supérieur (strictement) à zéro peut accepter des entités en tant que fils et peut les associer au réseau en leur donnant des adresses de sa plage d'adresses. Si $Cskip(d)$ est inférieur ou égal à zéro, le nœud ne peut pas avoir de fils et doit être une feuille (au niveau de la distribution des adresses).

Si on note A_{parent} l'adresse d'un coordinateur père de profondeur d, l'adresse A_k de son k-ème fils coordinateur est calculée selon la formule suivante :

$$A_k = A_{parent} + Cskip(d) \cdot (k-1) + 1,$$

et l'adresse A_n de sa n-ème feuille est calculée suivant la formule suivante :

$$A_n = A_{parent} + Cskip(d) \cdot R_m + n.$$

La figure 1.8 montre l'allocation par DAAM sur un exemple de topologie en arbre. Les paramètres du réseau sont fixés aux valeurs suivantes : $C_m = 3$, $R_m = 2$ et $L_m = 3$. Les liens représentent les communications père-fils dans le réseau. Le coordinateur du PAN a l'adresse 0. Dans cet exemple, nous avons $Cskip(0) = 10$. Le coordinateur du PAN alloue à son premier fils coordinateur l'adresse 1 et la plage d'adresses $[1; 10]$, et alloue à son second fils coordinateur l'adresse 11 et la plage d'adresse $[11; 20]$. À la première (et seule, puisque $C_m - R_m = 1$) feuille du coordinateur du PAN est donnée l'adresse $0 + 2.Cskip(0) + 1 = 21$. Les autres nœuds réalisent l'affectation d'adresses de la même manière. Notons que cette topologie correspond à la topologie avec le plus grand nombre de nœuds pour ces valeurs des paramètres du réseau.

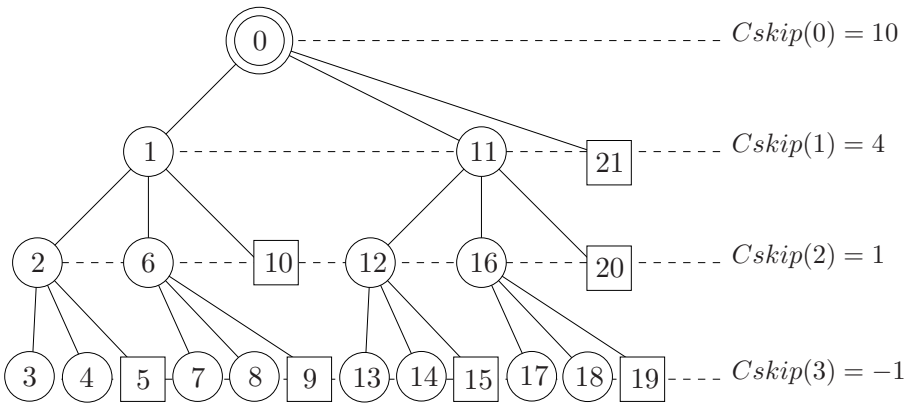


Figure 1.8 – Allocation des adresses des nœuds dans une topologie en arbre, avec $C_m = 3$, $R_m = 2$ et $L_m = 3$.

Le protocole de routage hiérarchique est basé sur la manière dont DAAM alloue les adresses.

Les seuls liens utilisés par le protocole de routage hiérarchique pour acheminer un paquet sont les liens de l'arbre. Les feuilles envoient toujours les paquets à leurs pères coordinateurs. Quand un coordinateur reçoit un paquet pour une destination, il vérifie si la destination appartient à sa plage d'adresses. Si c'est le cas, la destination est un descendant de ce coordinateur : le coordinateur détermine donc à quel fils la destination appartient, et envoie le paquet au fils concerné. Si la destination n'appartient pas à la plage d'adresses du coordinateur, le coordinateur envoie le paquet à son père.

Plus formellement, pour un coordinateur qui a l'adresse courte A et qui est à la profondeur d, le nœud destination ayant l'adresse courte D est un descendant du coordinateur A si l'on a :

$$A < D < A + Cskip(d-1).$$

Dans ce cas, le coordinateur A vérifie si le nœud destination est l'une de ses feuilles en déterminant si l'on a :

$$D > A + Rm \cdot Cskip(d).$$

Si cette formule n'est pas vraie, le nœud destination n'est pas une feuille. L'adresse du nœud fils coordinateur vers lequel le coordinateur A route le paquet est donnée comme suit :

$$A + 1 + \lfloor \frac{D-(A+1)}{Cskip(d)} \rfloor \cdot Cskip(d).$$

La figure 1.9 reprend l'exemple illustré dans la figure 1.8. Les flèches représentent le chemin de la source, qui est la feuille 20, vers la destination, qui est le coordinateur 7. La feuille 20 envoie le paquet à son père, qui est le nœud 11. Le coordinateur 11 remarque que l'adresse de la destination finale n'appartient pas à sa plage d'adresse $[11; 20]$, et envoie donc le paquet à son père, le coordinateur du PAN, qui a l'adresse 0. Le coordinateur du PAN doit déterminer si la destination 7 appartient à sa plage d'adresses $[0; 21]$. Puisque c'est le cas, le coordinateur du PAN doit déterminer si la destination est l'une de ses feuilles, ou si le paquet doit être envoyé à un fils coordinateur intermédiaire. Ici, le coordinateur 0 détecte que la destination 7 appartient à la plage d'adresse $[1; 10]$ de son fils coordinateur 1. Donc, le coordinateur du PAN achemine le paquet au coordinateur 1. De même, le coordinateur 1 détermine que l'adresse 7 appartient à la plage d'adresse $[6; 9]$ de son fils coordinateur 6. Le coordinateur 1 envoie ainsi le paquet au coordinateur 6. Finalement, le coordinateur 6 détermine que la destination 7 est l'un de ses coordinateurs fils, et lui envoie directement le paquet.

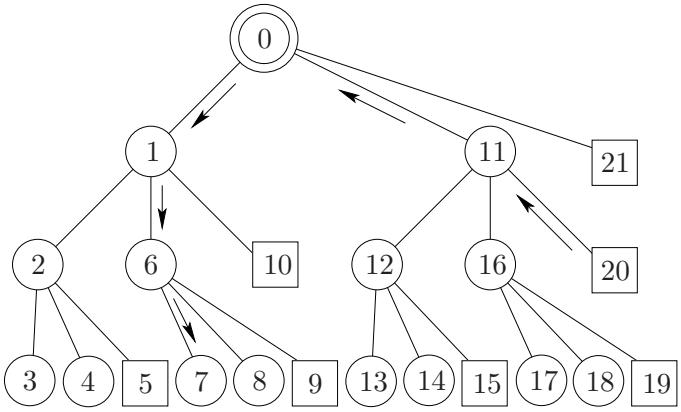


Figure 1.9 – Représentation du chemin de routage de la source 20 à la destination 7.

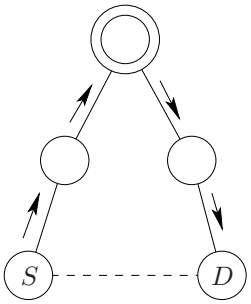


Figure 1.10 – Problème du protocole de routage hiérarchique.

Avantages et inconvénients du protocole de routage hiérarchique. Les principaux avantages du protocole de routage hiérarchique sont sa simplicité et son besoin limité en ressources. En effet, il ne nécessite aucun échange d'information entre les nœuds du réseau, pourvu que DAAM soit utilisé et que la topologie soit stable. Cependant, le principal inconvénient de ce protocole est le fait que les chemins de routage ne sont pas optimaux. En effet, le protocole de routage hiérarchique utilise seulement les relations père-fils et ignore les nœuds voisins. Par exemple, les paquets peuvent être acheminés à la destination en suivant plusieurs sauts, même si la destination se trouve à portée de la source. La figure 1.10 montre un exemple de ce problème. Les lignes pleines représentent les associations père-fils de l'arbre et les lignes pointillées représentent la portée des nœuds. Le paquet du nœud source S remonte l'arbre jusqu'au coordinateur du PAN en suivant les liens père-fils, et descend jusqu'à la destination D. Dans ce cas, quatre sauts sont nécessaires pour arriver à la destination. Cependant, si S avait envoyé le paquet directement à D, la transmission n'aurait nécessité qu'un seul saut.

1.4 Dimensionnement du découpage temporel

Les méthodes dénommées accès multiples à répartition dans le temps (AMRT) consistent à diviser le temps en intervalles de temps. Chaque intervalle de temps a la responsabilité d'offrir aux nœuds de bonnes conditions d'accès au médium. Le découpage temporel, offre une solution au problème d'économie d'énergie pour les réseaux de capteurs sans fil. Un bon découpage temporel conduit à limiter les collisions.

Le dimensionnement des intervalles de temps est relié au type de trafic généré par l'application et aux QoS requises. En effet, pour une application générant du trafic en rafale, les intervalles de temps doivent être suffisamment larges pour pouvoir supporter ce trafic. Mais si les intervalles de temps sont trop grands, du temps est gaspillé quand tout le trafic a été envoyé. Le découpage temporel peut être dimensionné d'une manière statique ou bien d'une manière dynamique. Dans cette section, nous détaillons ces deux modes de dimensionnement et nous montrons leurs avantages ainsi que leurs inconvénients.

1.4.1 Dimensionnement statique du découpage temporel

Le dimensionnement temporel statique dans un réseau de capteurs sans fil consiste à diviser le temps en périodes (comme la supertrame de IEEE 802.15.4). Chaque période est réservée pour un ensemble de nœuds dans le réseau. Un nœud ne peut pas accéder au canal pendant un intervalle de temps réservé à d'autres nœuds.

Un exemple de dimensionnement statique est le TDMA (pour *Time Division Multiple Access*) où chaque nœud possède son intervalle de temps. Un tel TDMA réduit la consommation d'énergie et évite les collisions. Cependant, l'attribution des intervalles de temps TDMA étant statique, si un nœud se voit attribuer un intervalle de temps et qu'il n'a pas de données à transmettre, il ne peut pas laisser cet intervalle de temps à un autre nœud. Cette méthode gaspille la bande passante, ce qui diminue le débit et augmente les délais.

Le découpage temporel statique sert à répondre aux besoins des QoS de plusieurs applications. La plupart des techniques d'optimisation des réseaux de capteurs est basée sur le découpage temporel. Comme nous l'avons déjà mentionné, pour le standard IEEE 802.15.4, la supertrame est divisée en une période obligatoire CAP et une autre optionnelle CFP, suivie d'une période d'inactivité si l'on veut économiser de l'énergie. Dans la CAP, le mécanisme CSMA/CA slotté est utilisé et dans la CFP, quand elle existe, le mécanisme TDMA est actif. Chacun de ces deux mécanismes est indispensable pour un type de trafic. En effet, pour une ap-

plication souhaitant assurer la bonne réception des données indépendamment du délai, TDMA est suggéré. Par contre, pour des applications plus réactives mais moins sensibles aux pertes, le mécanisme CSMA/CA slotté est suggéré.

1.4.2 Dimensionnement dynamique du découpage temporel

Le dimensionnement dynamique du découpage temporel consiste à adapter les intervalles de temps aux applications. La durée ainsi que l'allocation des intervalles de temps changent durant les communications selon les conditions et les changements du réseau. Dans ce qui suit, nous détaillons quelques travaux de recherche effectués sur le dimensionnement dynamique.

Dans [PRH02], les auteurs présentent une approche unifiée de réservation des intervalles de temps en TDMA avec un routage distribué, dans un réseau multi-sauts. La réservation dynamique des intervalles de temps TDMA fournit des informations sur la connectivité du réseau. Ces informations peuvent être utilisées pour réaliser un protocole de routage distribué capable de tolérer l'évolution de la connectivité du réseau. Dans la réservation dynamique des intervalles de temps, le temps est divisé en trames et les trames en des petits intervalles de temps. Une portion des ces intervalles de temps est utilisée pour diffuser un nombre suffisant de requêtes de réservation. Le but de cette méthode est de fournir à chaque nœud les informations suffisantes sur l'utilisation des intervalles de temps par son voisinage (voisins jusqu'à deux sauts). Ces informations permettent au nœud de choisir d'une manière efficace l'intervalle de temps pour qu'il transmette ses données. De plus, ces informations peuvent être utilisées pour construire les tables de routage. Cette approche unifiée est une conception inter-couches *(cross-layering)* entre la couche réseau et la couche accès au médium.

Dans [KUHN03], les auteurs proposent un protocole de réservation des intervalles de temps afin d'améliorer l'utilisation du canal. Ce protocole de réservation consiste à contrôler l'augmentation excessive des intervalles de temps non réservés en changeant la taille de la trame (le nombre des intervalles de temps dans la trame), d'une façon dynamique, selon le nombre de nœuds qui se trouvent dans la zone de contention. Les intervalles de temps non réservés sont attribués aux nœuds qui viennent de joindre le réseau. S'il n'y a aucun intervalle de temps disponible, le mécanisme de réservation crée des intervalles de temps non réservés en récupérant un intervalle de temps parmi ceux qui sont déjà attribués pour un autre nœud ou bien en agrandissant la taille des trames.

Dans [TCC09], les auteurs présentent des techniques adaptatives pour ajuster d'une manière dynamique la réservation des intervalles de temps TDMA dans les réseaux de capteurs sans fil.

Ces techniques sont capables de tracer les changements dans le réseau (en termes de trafic et de contention). Ces auteurs proposent un compromis entre les performances, d'une part, et l'augmentation des messages de signalement d'autre part. Ils ont montré que la réservation dynamique des intervalles de temps TDMA permet de s'adapter aux différentes conditions de trafic.

Le dimensionnement temporel dynamique a tendance à être complexe, peu flexible et présente des problèmes lors du passage à l'échelle [RWAS08]. En effet, dans un réseau de capteurs, le changement des conditions de propagation et ceux des liens sont fréquents (conditions variables du canal, défaillances des nœuds, ...).

1.5 *Cross-layering*

En plus des techniques de dimensionnement du découpage temporel, des protocoles de routage et des méthodes d'accès adaptés aux réseaux de capteurs sans fil, les techniques de *cross-layering* peuvent encore améliorer les performances du réseau [BGV05].

Dans cette partie, nous mentionnons les utilités du mécanisme de *cross-layering* et ses avantages par rapport au modèle OSI dans les réseaux de capteurs sans fil. Nous montrons les différents type de *cross-layering* qu'une solution réseau peut offrir et nous concentrons notre étude sur le *cross-layering* entre la couche réseau et la sous-couche MAC.

1.5.1 Modèle OSI et *Cross-layering*

Le modèle OSI (pour *Open System Interconnection*) est un modèle conceptuel décrivant les fonctionnalités nécessaires à la communication et à l'organisation de ces fonctions. Le modèle OSI comporte sept couches représentées dans la figure 1.11. Ces couches fonctionnent indépendamment les unes des autres. Elles sont parfois réparties en deux groupes. Les quatre couches inférieures sont plutôt orientées communication, les couches transport et réseau sont fournies par un système d'exploitation et les couches physique et liaison de données sont fournies par le matériel. Les trois couches supérieures sont plutôt orientées application et plutôt réalisées par des programmes spécifiques. Dans le monde IP ainsi que dans le monde réseau de capteurs sans fils, ces trois couches sont rarement distinguées. Dans ce cas, toutes les fonctions de ces couches sont considérées comme partie intégrante du protocole applicatif.

Les caractéristiques de chaque couche du modèle OSI sont les suivantes.

1. La couche physique est chargée de la transmission effective des signaux entre les interlo-

couche application
couche présentation
couche session
couche transport
couche réseau
couche liaison de données
couche physique

Figure 1.11 – Les sept couches du modèle OSI.

cuteurs. Son service est limité à l'émission et à la réception d'un bit ou d'un train de bit continu (notamment pour les supports synchrones).

2. La couche liaison de données gère les communications entre entités adjacentes, directement reliées entre elles par un support physique (en gérant les erreurs).
3. La couche réseau gère les communications de proche en proche, généralement entre les entités : routage et adressage.
4. La couche transport gère les communications de bout en bout entre processus.
5. La couche session gère la synchronisation des échanges et les transactions de bout-en-bout. Elle permet l'ouverture et la fermeture de session.
6. La couche présentation est chargée du codage des données applicatives, précisément de la conversion entre données manipulées au niveau applicatif et chaînes d'octets effectivement transmises.
7. La couche application est le point d'accès aux services réseaux.

Bien que le principe de séparation des couches se soit avéré efficace pour les réseaux filaires, il ne constitue pas une solution optimale pour les réseaux sans fil, en particulier pour les réseaux de capteurs sans fil. En effet, les spécificités des réseaux de capteurs sans fil et des réseaux *ad hoc* incluent :

- des changements dynamiques des conditions du réseau en raison de la mobilité ou de la versatilité des conditions de propagation du médium physique,
- des ressources limitées en capacité (comme la bande passante, la quantité d'énergie, la mémoire limitée),
- la présence d'interférences radio.

Ces spécificités apportent de nouvelles contraintes dans la conception des protocoles pour les réseaux de capteurs sans fil. Ainsi, ces protocoles doivent être économes en énergie et adaptatifs. De plus, ils doivent satisfaire les QoS désirées par les applications. Pour répondre à toutes ces contraintes, le mécanisme de *cross-layering* qui assure l'optimisation entre les différentes couches de la pile protocolaire est nécessaire. La figure 1.12 illustre les communications principales utilisées pour le *cross-layering* (ces communications sont illustrées par des flèches à double sens). Nous pouvons remarquer sur la figure que même si les couches sont adjacentes, l'optimisation du *cross-layering* peut avoir lieu.

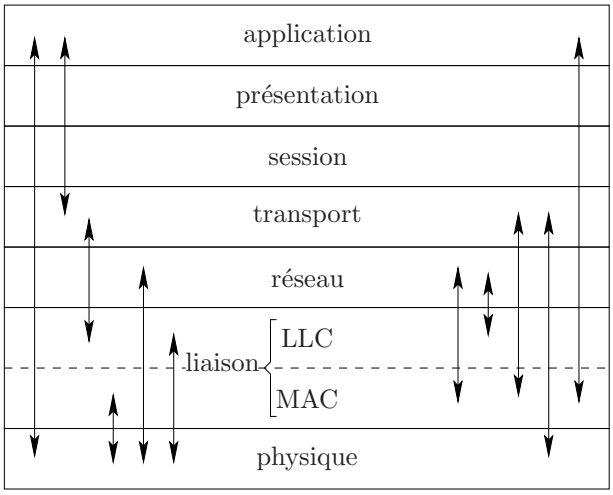


Figure 1.12 – *Cross-layering* envisageable entre les différentes couches de la pile protocolaire du modèle OSI [Gav06].

Chaque couche dans un réseau dépend des autres couches. Dans les réseaux sans fil, la couche physique doit s'adapter aux changements rapides des caractéristiques des liens. La sous-couche MAC a intérêt à minimiser les collisions. La couche réseau est responsable de déterminer et de distribuer l'information utilisée pour calculer les chemins, et de maintenir l'efficacité du réseau malgré les changements des liens et la variation de la bande passante.

Les paramètres pertinents du *cross-layering* dépendent des contraintes et des objectifs des applications. Ces paramètres peuvent inclure [RI04] :

- des informations sur l'état du canal, comme l'estimation de la réponse du canal en temps et en domaine de fréquence, la puissance du signal, le niveau d'interférences, le modèle des interférences ;
- des paramètres reliés à la QoS, comme le délai, le taux de pertes, le débit ;
- des informations sur les ressources disponibles, comme le nombre et le type des antennes, le niveau d'énergie résiduelle des batteries ;

– le modèle du trafic offert au réseau (statistiques sur les données produites), la connaissance de taux de données produites, la connaissance de la fragmentation des données, la taille des paquets, la taille des files d'attente.

Dans cette thèse, nous concentrons notre étude sur le *cross-layering* entre la sous-couche MAC et la couche réseau. Dans ce qui suit, nous détaillons quelques recherches effectuées sur ce type de *cross-layering*.

1.5.2 *Cross-layering* MAC/réseau

Généralement, les protocoles de routage sont basés sur l'émetteur, c'est-à-dire que c'est l'émetteur qui choisit le prochain saut en fonction de sa connaissance du voisinage. Si les informations du voisinage ne sont pas à jour, la décision prise par un protocole de routage basé sur l'émetteur peut être inadaptée. Dans les réseaux sans fil, il existe de nombreux protocoles de routage basés sur le récepteur, c'est-à-dire que tous les voisins d'un émetteur reçoivent le même paquet et s'organisent pour qu'un seul d'entre eux achemine le paquet. L'avantage de ces protocoles est qu'ils sont plus robustes aux changements de topologie. Plus de détails sur ce type de protocoles sont dans [MML09].

Dans plusieurs travaux [SAB04, ZR03b, ZR03a], un protocole de routage basé sur le récepteur est exploité pour le *cross-layering* entre la sous-couche MAC et la couche réseau. Dans l'approche *cross-layering* utilisant le protocole de routage basé sur le récepteur, le prochain saut est choisi selon la contention du voisinage. Dans [ZR03b, ZR03a], les auteurs étudient la consommation d'énergie, la latence et la performance multi-sauts du protocole de routage proposé.

Dans [SAB04], les auteurs proposent un protocole de routage passif distribué (DPRD, pour *Distributed Passive Routing Decisions*). Ce protocole est une approche combinée utilisant les fonctionnalités de routage ainsi que celles de la méthode d'accès. Cette méthode d'accès réduit la consommation d'énergie en minimisant les messages de signalement sur les décisions de routage qui est basé sur le récepteur. De plus, la latence de ce protocole de routage basé sur le récepteur est présentée en se basant sur différentes fonctions de délais et des taux de collisions.

De même, dans [Fer05], les auteurs proposent un protocole intégré MAC et routage, appelé MACRO. Ce protocole exploite la capacité des capteurs afin d'ajuster leur puissance de transmission. Afin de sélectionner le nœud relais suivant, une compétition est déclenchée à chaque saut. Le nœud ayant l'énergie résiduelle la plus grande est choisi. Ceci est réalisé grâce à la maximisation d'un nouveau paramètre qui représente le progrès vers la destination par unité de puissance transmise.

Un mécanisme d'ordonnancement et de routage a été proposé dans [Sic04] pour un trafic périodique dans un réseau de capteurs sans fil. Les nœuds forment un ordonnancement *on-off* au niveau MAC, et les routes sont établies de sorte que les nœuds soient réveillés seulement quand nécessaire. Puisque le trafic est périodique, les ordonnancements (au niveau MAC) sont alors maintenus pour favoriser au réseau une consommation d'énergie minimale.

Dans [BBB09], les auteurs proposent une technique de *cross-layering* intégrant la couche réseau et la sous-couche MAC. Au niveau de la couche réseau, ils proposent d'équilibrer le trafic dans le réseau de capteurs sans fil. Ils montrent que la transmission du trafic généré par chaque capteur en utilisant des chemins multiples permet de conserver l'énergie plus significativement que lorsqu'un seul chemin de routage est utilisé. D'autre part, au niveau MAC, les auteurs proposent de contrôler le nombre de retransmissions au niveau MAC sur chaque lien sans fil pour économiser de l'énergie.

Dans [Mah09], les auteurs proposent une nouvelle technique de *cross-layering* entre la couche réseau et la sous-couche MAC dans un réseau de capteurs sans fil. Cette technique est dénommée SERENA. Dans SERENA, un nœud détermine quand il doit être réveillé et quand il peut passer en mode sommeil. Ceci est complètement transparent pour la couche réseau, dans la mesure où un nœud est réveillé chaque fois qu'un message est envoyé. SERENA colorie les nœuds du réseau. La sous-couche MAC réserve un (ou plusieurs) intervalle(s) de temps par couleur. Durant un intervalle de temps, tous les nœuds ayant la couleur correspondante peuvent transmettre tant que leurs prochains sauts sont réveillés. Les autres nœuds peuvent dormir. Afin de rendre cela possible, SERENA doit notifier la sous-couche MAC de la couleur du nœud considéré ainsi que des couleurs de ses voisins à un saut. Pour bien dimensionner la trame, la sous-couche MAC doit connaître le nombre maximal de couleurs utilisées dans le réseau. Les nœuds doivent alors se comporter selon les périodes d'activités calculées par la sous-couche MAC en fonction des couleurs données par SERENA.

1.6 Conclusion

Dans ce chapitre, nous avons détaillé les standards sur lesquels nous nous sommes basés pour réaliser cette thèse. Tout d'abord, nous avons décrit le standard IEEE 802.15.4 qui définit les couches basses de la pile protocolaire d'un réseau de capteurs sans fil. Ensuite, nous avons détaillé le standard ZigBee qui définit les couches hautes d'un réseau de capteurs sans fil. Ces deux standards répondent convenablement à la contrainte énergétique des réseaux de capteurs

sans fil. En effet, ils définissent au niveau MAC une période d'inactivité durant laquelle les nœuds peuvent se mettre en mode sommeil pour économiser de l'énergie.

Ensuite, nous avons détaillé les principes de découpages temporels sur lesquels sont basés les standards ainsi que la plupart des protocoles pour les réseaux de capteurs sans fil. Nous avons montré leurs avantages et leurs inconvénients. Nous avons ensuite détaillé la technique du *cross-layering* qui permet d'optimiser le réseau et améliorer ses performances en termes de délai, de débit, et de taux de pertes, sans augmenter la consommation d'énergie.

Chapitre 2

Cross-layering dans une architecture multi-couches

Les techniques de communications inter-couches, nommées *cross-layering*, sont très utilisées dans l'ingénierie des protocoles pour améliorer les performances des réseaux [Aso10]. Par ailleurs, de nombreux travaux considèrent les protocoles de routage et les méthodes d'accès simultanément [WBS09, Fer05]. Pour chaque combinaison d'un protocole de routage et d'un protocole MAC, plusieurs chercheurs [SRDV08, PRML06, KKL+06] identifient des types de trafic qui peuvent être acheminés d'une manière efficace. La conclusion partagée par la majorité de ces travaux est qu'une seule combinaison de protocoles de routage et MAC ne peut pas être adaptée à tous les besoins de qualité de service [AWW05, BPC+07].

Les réseaux de capteurs sans fil sont utilisés de nos jours dans plusieurs applications afin de surveiller l'environnement et de détecter des événements critiques. La tendance actuelle est de déployer un réseau de capteurs d'utilité globale, capable d'être utilisé par plusieurs applications simultanément [CHCP07]. Chaque application génère plusieurs types de trafic spécifiques (par exemple, du trafic périodique, des paquets de contrôle, des alarmes), associés à différents besoins de QoS.

Dans ce chapitre, nous proposons une architecture permettant d'assurer un grand nombre de QoS pour plusieurs applications déployées sur un même réseau de capteurs sans fil. Ensuite, nous proposons une méthode afin d'améliorer les performances de cette architecture.

2.1 Introduction

La plupart des réseaux de capteurs existants fonctionnent en utilisant un seul protocole de routage et une seule méthode d'accès. Cependant, cette combinaison de protocoles ne peut pas fournir une bonne performance pour toutes les QoS [AWW05, BPC+07, Aso10]. En effet, chaque combinaison est adaptée pour quelques types de trafic. Par exemple, certains protocoles peuvent fournir une grande réactivité aux événements d'alarmes, tandis que d'autres permettent de prolonger la durée de vie du réseau.

Une façon de fournir plusieurs QoS est de faire fonctionner plusieurs combinaisons, chacune constituée d'un protocole de routage $\mathcal{R}$ et d'une méthode d'accès $\mathcal{M}$, dans le même réseau de capteurs sans fil. Ceci peut être effectué en ordonnançant l'activité de tous les nœuds du réseau (au niveau de la sous-couche MAC), et en s'assurant qu'à chaque instant, une seule combinaison de protocoles est active pour tous les nœuds. Plus précisément, le temps est divisé en plusieurs périodes. Durant chaque période p_i, un protocole de routage $\mathcal{R}_i$ et une méthode d'accès $\mathcal{M}_i$ sont activés. Une étape de synchronisation précède la première période. L'étape de synchronisation a pour rôle d'assurer que tous les nœuds du réseau soient synchronisés et possèdent une vision commune des périodes p_i. Cette étape permet généralement de communiquer à tous les nœuds la durée de chaque période p_i.

Un tel mécanisme est utilisé dans le standard IEEE 802.15.4 [IEE06] où deux méthodes d'accès différentes sont utilisées : la méthode d'accès CSMA/CA slotté utilisée durant la période CAP, et les GTS (pour *Guaranteed Time Slot*) utilisés durant la période CFP, et dans la méthode d'accès MaCARI [CGM08] que nous décrivons plus tard.

Ordonnancer plusieurs combinaisons de protocoles afin d'avoir une différenciation de services aboutit à un problème de marquage et à un problème de dimensionnement. Le problème de marquage vient du fait que les paquets doivent être marqués par la couche application pour qu'ils puissent être livrés au protocole assurant la QoS requise ce qui peut revenir à la dédier à une combinaison $(\mathcal{M}_i, \mathcal{R}_i)$ particulière. Un paquet marqué par i peut seulement être envoyé durant la période p_i. Ceci assure que tous les nœuds utilisent le même protocole de routage pour le même paquet, et garantit que des décisions de routage cohérentes soient prises. Si la période en cours n'est pas p_i, un paquet marqué par i doit attendre la prochaine période p_i pour être envoyé, ce qui augmente le délai. Le problème de dimensionnement intervient dans le choix de la durée des périodes. Un dimensionnement statique n'est pas convenable pour un trafic arrivant en rafales (comme pour des alarmes). Si la période p_i est trop petite pour traiter

le trafic marqué par i, certains paquets doivent attendre l'occurrence de la prochaine période p_i pour être envoyés. Si la période p_i est trop grande pour traiter le trafic marqué par i, une partie du temps alloué pour p_i est perdue. Un dimensionnement dynamique requiert des algorithmes complexes et une surcharge significative en terme de messages de contrôle afin d'adapter les périodes à la génération du trafic.

Cette problématique constitue le cœur de cette thèse. Dans ce chapitre, nous proposons deux architectures permettant de gérer plusieurs combinaisons de protocoles. Nous décrivons tout d'abord une architecture où plusieurs protocoles coexistent dans la même couche, et nous montrons ses inconvénients. Ensuite, nous proposons une architecture où des échanges des paquets peuvent avoir lieu entre différentes combinaisons de protocoles. Nous nous concentrons alors sur l'occurrence de détours de routage dans le réseau pouvant survenir quand un paquet est routé selon plusieurs protocoles de routage. Plus précisément, nous étudions quand et comment permettre à des paquets d'être routés par plusieurs protocoles de routage. Notre approche peut être utilisée pour router des paquets marqués par i selon un protocole de routage $\mathcal{R}_j$ (avec $i \neq j$). Nous allons voir comment cette architecture peut réduire l'impact des deux problèmes que nous avons déjà cités.

2.2 Architecture avec plusieurs protocoles par couche

Dans cette partie, nous décrivons une nouvelle architecture protocolaire basée sur une approche multi-couches [ERGM11]. Nous utilisons le terme multi-couches pour indiquer la coexistence de plusieurs protocoles MAC ou de plusieurs protocoles de routage.

La figure 2.1 montre l'architecture multi-couches que nous proposons. Cette architecture peut être composée de plusieurs combinaisons de protocoles $(\mathcal{M}_i, \mathcal{R}_i)$, avec une file d'attente $\mathcal{Q}_i$ associée à chaque combinaison. Chaque combinaison est dédiée à un ensemble restreint de types de trafic.

2.2.1 Description fonctionnelle de l'architecture multi-couches

À notre connaissance, il n'existe pas d'autre architecture où plusieurs protocoles de routage ou plusieurs protocoles d'accès indépendants peuvent coexister. Quelques architectures permettent la coexistence de plusieurs méthodes d'accès (comme IEEE 802.15.4 et ZigBee), mais ces architectures ne sont pas capables d'intégrer des protocoles quelconques.

La mise en place de l'architecture multi-couches nécessite la répétition d'un ordonnancement

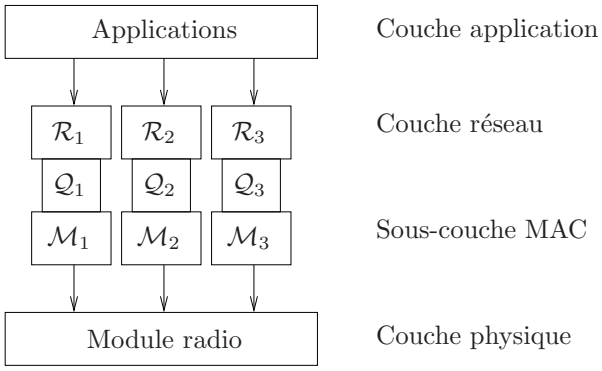


Figure 2.1 – Pile protocolaire de l'architecture multi-couches.

de périodes que nous appelons cycle. Durant chaque période p_i, un protocole MAC $\mathcal{M}_i$ et un protocole de routage $\mathcal{R}_i$ sont activés. Pour assurer la cohérence de l'accès au médium et du routage, nous supposons que tous les noeuds sont synchronisés et se trouvent à la même période à tout instant.

Deux paramètres majeurs doivent être partagés entre les noeuds du réseau : l'instant de démarrage et de fin de chaque période, ainsi que les protocoles qui doivent être activés dans chaque période. Pour partager ces informations, nous supposons qu'il existe un mécanisme de synchronisation qui précède la première période. Ce mécanisme de synchronisation peut être répété au début de chaque cycle.

La figure 2.2 montre un exemple de cycle avec une période de synchronisation et deux combinaisons de protocoles.

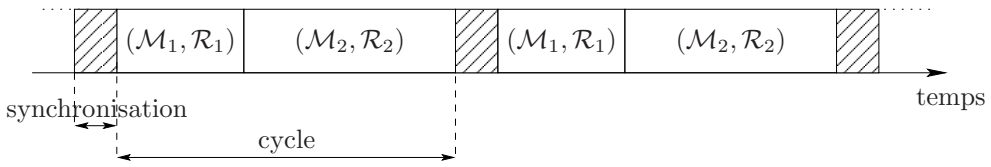


Figure 2.2 – Un exemple de cycle dans notre architecture multi-couches.

Nous considérons que chaque paquet est marqué par la couche application (ou par une couche qui gère la QoS). Comme première hypothèse, un paquet marqué par i peut être traité uniquement durant la période p_i.

La suite de cette partie décrit une implémentation centralisée de notre architecture. Cette approche centralisée a été choisie afin de simplifier la description. L'architecture peut être distribuée pourvu que le mécanisme de synchronisation le soit aussi.

2.2.2 Description technique de l'architecture multi-couches

Dans cette partie, nous détaillons d'une manière technique les différentes étapes à suivre et les différentes conditions à respecter afin d'implémenter une architecture multi-couches. Tout d'abord, nous décrivons l'étape de synchronisation, qui est indispensable au réseau global afin de bien gérer le fonctionnement des noeuds. Ensuite, nous décrivons l'étape d'ordonnancement des périodes, qui dépend des contraintes imposées par la sous-couche MAC et la couche réseau. Enfin, nous terminons par des spécifications techniques qui mettent en relief la procédure d'envoi des paquets durant une période quelconque.

Synchronisation

Le mécanisme de synchronisation a un double rôle : il assure que tous les noeuds du réseau connaissent l'instant du début de la première période du cycle et il permet à tous les noeuds du réseau de connaître l'ordonnancement des périodes.

Plusieurs protocoles existants peuvent être utilisés pour réaliser cette synchronisation. On peut mentionner par exemple TPSN (pour *Timing-Sync Protocol for Sensor Networks*) [GKS03], ou bien MaCARI [CGM08].

Ordonnancement des périodes

L'ordonnancement des périodes d'une architecture multi-couches est construit (généralement hors ligne) selon plusieurs contraintes.

1. Les protocoles MAC et les protocoles de routage ainsi que la durée de chaque période, dépendent de la production du trafic attendue, des propriétés des protocoles et de la QoS requise pour chaque trafic. En effet, chaque combinaison de protocoles doit être choisie en tenant compte des caractéristiques de ces protocoles et des QoS requises pour le trafic. Par exemple, s'il existe un trafic périodique avec des besoins faibles de QoS, et un trafic contraint par le temps de traversée du réseau qui doit être rapidement acheminé, il est intéressant d'avoir deux combinaisons de protocoles de routage et de MAC l'une pour le trafic périodique et l'autre pour le trafic contraint. La durée de chaque période dépend aussi de la génération du trafic. Si les deux tiers du trafic possèdent un besoin faible de QoS et le tiers restant du trafic nécessite une QoS élevée, il est raisonnable de réserver deux tiers du temps pour la combinaison de protocoles convenables pour le trafic à faible QoS et allouer un tiers du temps pour la combinaison de protocoles convenables pour le

trafic de QoS élevée.

2. Chaque protocole MAC nécessite une précision de synchronisation différente. Donc, les protocoles MAC ayant des besoins de synchronisation fine doivent être ordonnancés avant ceux ayant des besoins de synchronisation plus faible. En effet, la synchronisation des noeuds se dégrade à mesure que le temps avance dans un cycle. Chaque protocole MAC requiert une qualité de synchronisation différente. Le problème principal de la synchronisation est que la dérive d'horloge de chaque noeud rend quelques protocoles MAC inapplicables après une longue durée sans synchronisation. Si nous considérons ε l'erreur de synchronisation initiale à l'instant t et Δ la dérive d'horloge maximale, la désynchronisation maximale entre deux noeuds au temps $t+\delta$ est $\varepsilon+2\Delta\delta$. Si δ_i est l'instant à partir duquel $\varepsilon+2\Delta\delta_i$ dépasse la désynchronisation maximale tolérée par le protocole MAC $\mathcal{M}_i$, ce protocole MAC $\mathcal{M}_i$ doit être ordonnancé de manière à ce que sa période dans l'ordonnancement se termine avant le moment $t+\delta_i$. Par exemple, le mécanisme CSMA/CA slotté requiert des noeuds qu'ils soient synchronisés pour qu'ils aient une vision similaire (mais pas nécessairement parfaite) de chaque période de *backoff* de 320 μs. Même si nous supposons une synchronisation parfaite des noeuds à la fin de la synchronisation (c'est-à-dire, $\varepsilon=0$), avec une dérive d'horloge maximale de $\Delta=40$ ppm, la désynchronisation maximale entre deux noeuds atteint 320 μs après 4 secondes (c'est à dire, quand $\delta=4$).
3. Chaque protocole de routage routant indépendemment les paquets, il faut s'assurer qu'il n'y a pas de détours (au sens allongement du trajet risquant d'augmenter le temps de trajet et la charge du réseau) dus au routage quand plusieurs protocoles de routage sont combinés. L'apparition des détours et leur évitement est décrit dans 2.3.3 et dans 2.4 respectivement. Chaque protocole de routage route les paquets indépendemment les uns des autres : si plusieurs protocoles de routage sont utilisés pour acheminer un même paquet, il est possible d'avoir des détours entre les protocoles utilisés, (même si chaque protocole pris indépendemment évite les boucles). Dans la suite de ce chapitre, nous expliquons comment et pourquoi de tels détours peuvent apparaître dans le réseau.

Envoi des paquets

Dans notre architecture, une file d'attente spécifique $\mathcal{Q}_i$ est utilisée pour chaque combinaison $(\mathcal{M}_i, \mathcal{R}_i)$. Cette file d'attente $\mathcal{Q}_i$ reçoit seulement des paquets marqués par i.

Le marquage des paquets n'entraîne pas de surcharge significative en termes de taille du paquet. L'application marque les paquets et l'envoie à la couche réseau qui à son tour calcule

le prochain saut en utilisant le protocole de routage $\mathcal{R}_i$ et le stocke dans la file d'attente $\mathcal{Q}_i$ en attente d'être traité au niveau MAC par le protocole MAC $\mathcal{M}_i$.

2.2.3 Problème de dimensionnement

Rappelons qu'un dimensionnement statique des périodes n'est pas convenable pour des changements fréquents des conditions du trafic, ou pour un trafic en rafale. Quand une période est trop petite, les paquets en surplus doivent attendre l'activation de la prochaine période, ce qui augmente les délais. Quand la période est trop grande, du temps est perdu quand tous les paquets ont été envoyés. Un dimensionnement dynamique nécessite des algorithmes sophistiqués et un grand nombre de messages de contrôle afin de pouvoir adapter la durée des périodes au trafic généré.

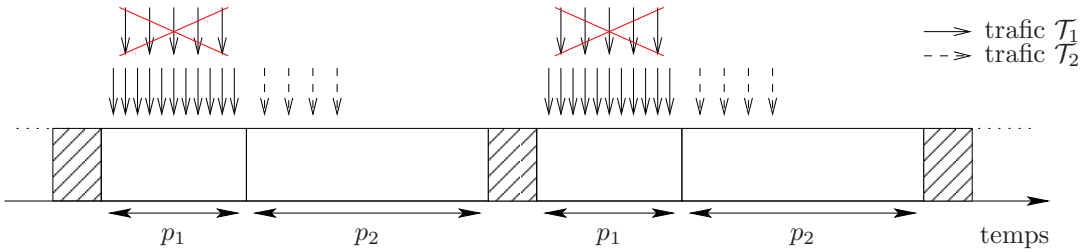


Figure 2.3 – Exemple d'un mauvais dimensionnement. La période p_1 ne peut pas traiter tous les paquets du trafic $\mathcal{T}_1$. La période p_2 traite tous les paquets du trafic $\mathcal{T}_2$ et le temps restant est gaspillé.

La figure 2.3 montre l'inconvénient du dimensionnement statique des périodes. Dans cet exemple, on suppose que le cycle, délimité par deux périodes de synchronisation, est divisé en deux périodes : la période p_1 dédiée au trafic $\mathcal{T}_1$ et la période p_2 dédiée au trafic $\mathcal{T}_2$. Le dimensionnement de ces deux périodes n'est pas optimal. En effet, la période p_1 n'arrive pas à traiter tout le trafic $\mathcal{T}_1$ (représenté par des flèches pleines sur la figure) et les trames qui sont générées vers la fin de la période p_1 n'ont pas le temps d'être envoyées. Les trames doivent attendre la prochaine occurrence de p_1. De même, la période p_2 est trop grande : le trafic $\mathcal{T}_2$ (représenté par des flèches pointillées) étant complètement envoyé durant p_2, le temps restant de p_2 est gaspillé.

2.3 Échanges des files d'attente

Pour régler le problème de dimensionnement des périodes, nous proposons un algorithme d'échanges des paquets entre les files d'attente de notre architecture [ERCGM11]. Notre but est

de permettre à une combinaison $(\mathcal{M}_i, \mathcal{R}_i)$ d'extraire et de traiter des paquets de la file d'attente $\mathcal{Q}_j$, avec $j \neq i$. Si $\mathcal{Q}_i$ est vide durant la période p_i, le reste de la période p_i peut être utilisé pour acheminer des paquets d'autres périodes.

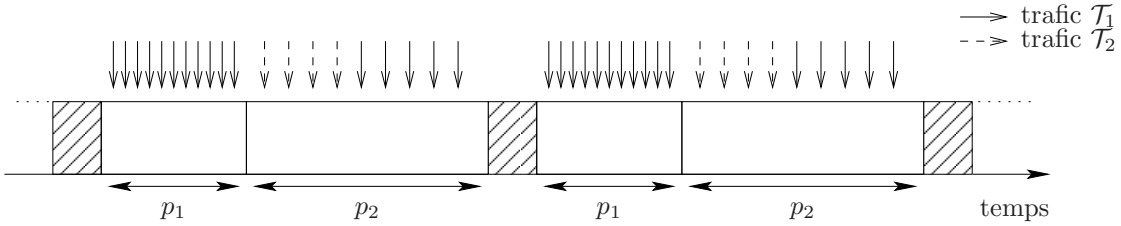


Figure 2.4 – Solution au problème de dimensionnement utilisant des échanges de paquets entre les files d'attente. Le temps restant de la période p_2 est utilisé pour traiter la surcharge du trafic $\mathcal{T}_1$ de la période p_1.

La figure 2.4 montre notre solution au problème de dimensionnement. Avec ce mécanisme que nous proposons, la période p_2 est capable de traiter et d'envoyer des paquets du trafic $\mathcal{T}_1$ (correspondant à la première QoS). Pour éviter de pénaliser le trafic $\mathcal{T}_2$ (correspondant à la deuxième QoS), ce mécanisme n'est appliqué que lorsque la période p_2 n'a plus de paquets de cette deuxième QoS à envoyer. Avec ce mécanisme, certains paquets n'ont plus à attendre la prochaine occurrence de leur période pour être traités.

Dans le reste de cette partie, nous décrivons le mécanisme des échanges entre les files d'attente de l'architecture multi-couches. Nous donnons, ensuite les détails techniques qui permettent de résoudre en pratique le problème de dimensionnement des périodes. Enfin, nous détaillons le problème d'apparition des détours que peut induire ce mécanisme d'échanges.

2.3.1 Description du mécanisme

Le mécanisme que nous proposons dans cette partie consiste à échanger les paquets générés par les applications entre les combinaisons $(\mathcal{M}_i, \mathcal{R}_i)$ coexistantes dans le réseau. Avec ce mécanisme, les paquets marqués par i et gérés habituellement par la combinaison $(\mathcal{M}_i, \mathcal{R}_i)$ pour être traités durant la période p_i, peuvent être gérés par la combinaison $(\mathcal{M}_j, \mathcal{R}_j)$ pour être traités durant la période p_j, avec $i \neq j$.

La figure 2.5 montre le traitement des paquets sans et avec échanges des files d'attente. Sans échange, c'est-à-dire quand un paquet marqué par i est traité seulement dans la période p_i, le temps est perdu quand $\mathcal{Q}_i$ est vide. Donc, tous les autres paquets qui ne sont pas marqués par i (c'est-à-dire marqués par j, $\forall j \neq i$), peuvent souffrir d'un grand délai. Ceci est montré sur la figure 2.5(a). La figure 2.5(b) montre comment les échanges des files d'attente permettent de

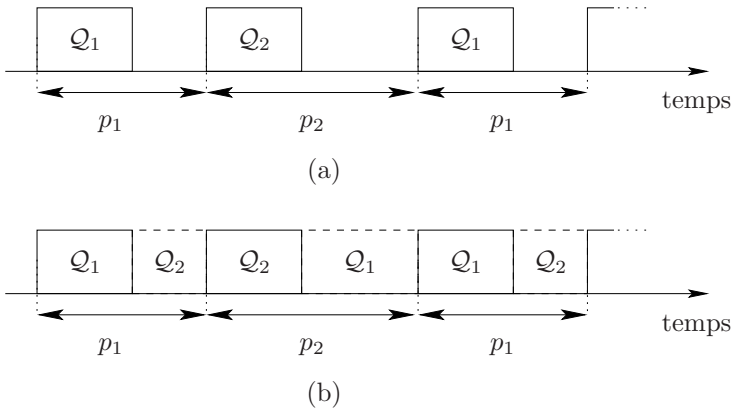


Figure 2.5 – Traitement des paquets : (a) sans échanges des paquets entre les files d'attente, (b) avec échanges des paquets entre les files d'attente.

profiter du temps non utilisé dans la période p_i pour transmettre des paquets marqués par j. Ainsi, le temps n'est plus gaspillé et les délais sont réduits. Notons qu'il est possible d'autoriser les échanges des paquets d'une file d'attente $\mathcal{Q}_i$ à une autre $\mathcal{Q}_j$ même si $\mathcal{Q}_i$ n'est pas vide. En effet, dans quelques cas (par exemple, quand les paquets de $\mathcal{Q}_j$ possèdent une priorité plus élevée que les paquets de $\mathcal{Q}_i$), il pourrait être bénéfique d'envoyer les paquets de plus haute priorité de $\mathcal{Q}_j$ avant ceux de $\mathcal{Q}_i$, même durant la période p_i.

2.3.2 Résolution du problème de dimensionnement

Avec notre approche, un dimensionnement fin pour les périodes n'est plus critique : si une période n'est pas suffisante pour les paquets qui lui sont dédiés, un autre protocole de routage peut acheminer les paquets restants. Si une période p_i est trop grande, le temps n'est pas perdu puisque la période est réutilisée pour acheminer des paquets dédiés à d'autres périodes.

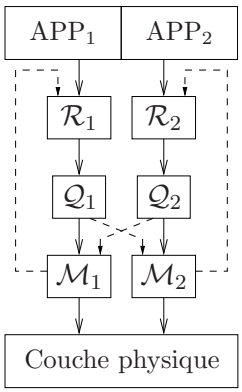


Figure 2.6 – Notre architecture autorise un paquet à être traité par tout protocole de routage.

Notre mécanisme permet les échanges des paquets entre les files d'attente du réseau, comme

montré sur la figure 2.6. Quand un protocole $\mathcal{M}_i$ a envoyé tous les paquets de sa file d'attente $\mathcal{Q}_i$, il peut extraire des paquets d'une autre file d'attente $\mathcal{Q}_j$ (avec $i \neq j$) afin d'utiliser ce qui reste de la période p_i. Cependant, seul le protocole de routage $\mathcal{R}_i$ peut être utilisé dans la période p_i. En effet, le prochain saut selon $\mathcal{R}_j$ peut être inactif pendant la période p_i. Le prochain saut du paquet marqué par j est recalculé selon $\mathcal{R}_i$ et envoyé par $\mathcal{M}_i$, sans changer le marquage du paquet (afin de conserver la QoS).

2.3.3 Apparition de détours

Une boucle correspond à la transmission en continu d'un paquet par une série de nœuds sans qu'il puisse atteindre sa destination finale. Une boucle cause de la congestion dans réseau puisque la bande passante est utilisée pour faire tourner le trafic en boucle. Nous définissons un détour par une augmentation de la longueur du chemin établi entre la source et la destination. En d'autre terme, le détour est créé dans le réseau quand un même paquet passe par un nœud pour plus qu'une fois avant d'arriver à sa destination.

Les protocoles de routage sont conçus pour veiller à ce qu'aucun détour dû au routage n'apparaisse. Cependant, cette propriété n'est valide que lorsque tous les nœuds du réseau opèrent selon le même protocole pour un paquet donné. Quand un paquet est transmis selon plusieurs protocoles de routage, des détours peuvent apparaître dans le réseau. Dans la suite, nous nous concentrons sur les protocoles de routage, car seuls les protocoles de routage causent l'apparition de détours dans le réseau.

La figure 2.7 décrit un exemple où un paquet fait un détour. Le nœud A a un paquet à envoyer au nœud E, $\mathcal{R}_1$ (représenté par des flèches pleines) et $\mathcal{R}_2$ (représenté par des flèches pointillées) sont alternés tous les deux sauts. Cette hypothèse permet de simplifier l'exemple. Toutefois, en pratique, une période de durée constante ne peut pas être traduite en un nombre de sauts fixe puisque ce nombre dépend de la méthode d'accès qui possède souvent un comportement dépendant beaucoup de la charge locale du réseau. Le scénario commence au début de la période p_1. A envoie le paquet à B, et B l'envoie à D. Ensuite, le protocole de routage change et $\mathcal{R}_2$ devient actif. D envoie le paquet à B, et B l'envoie à C. Ensuite, le protocole de routage actif redevient $\mathcal{R}_1$. C envoie le paquet à D, et D l'envoie à E. Le chemin suivi par le paquet est (A, B, D, B, C, D, E), ce qui correspond à six sauts. Notons que si seul $\mathcal{R}_1$ était utilisé, le chemin serait (A, B, D, E), ce qui correspond à trois sauts. Si seul $\mathcal{R}_2$ était utilisé, le chemin serait (A, C, E), ce qui correspond à deux sauts. L'utilisation de notre architecture a augmenté la distance parcourue par le paquet à cause de l'apparition d'un détour dans le réseau.

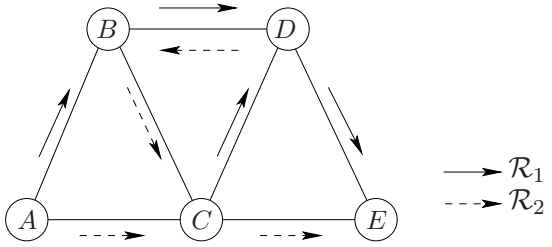


Figure 2.7 – Un détour apparaît dans le réseau quand A envoie un paquet à E, si les protocoles alternent chaque deux sauts et si le premier protocole de routage utilisé est $\mathcal{R}_1$. Dans ce cas, le chemin suivi par le paquet est (A, B, D, B, C, D, E), qui est plus long que les chemins correspondant à chaque protocole à part (ces chemins sont (A, B, D, E) pour $\mathcal{R}_1$ et (A, C, E) pour $\mathcal{R}_2$).

L'apparition de détours est un phénomène lié à la fois aux protocoles de routage (puisqu'ils décident des routes) et aux protocoles MAC (puisqu'ils influent sur le nombre de sauts parcourus par une trame pendant une période donnée et sur les pertes de trames dues à la purge ou aux collisions). Les longs détours ont peu de chances d'apparaître en pratique puisqu'ils sont dus à des alternances de protocoles de routage répétitives et particulières, appliquées à un même paquet. Cette répétitivité des alternances pour un même paquet est peu probable pour les raisons suivantes :

- Le temps que passe un paquet dans la file d'attente de chaque nœud est très variable (car ce temps dépend de sa position dans la file d'attente et du délai pour accéder au médium). Même si les protocoles de routage sont alternés de manière répétitives, il est peu probable qu'un même paquet subisse cette alternance répétitive.
- Certaines méthodes d'accès (comme CSMA/CA slotté) ne garantissent pas de transmettre n paquets dans un intervalle de temps donné. Même si la durée attribuée à chaque protocole de routage est la même, en fonction de la méthode d'accès sous-jacente, le nombre effectif de paquets envoyés durant cette période est variable, ce qui peut réduire le caractère répétitif des alternances.

En présence d'un grand nombre de détours, le réseau est surchargé, et les files d'attente des différents nœuds se remplissent de paquets rapidement. La purge de trames réduit la congestion dans le réseau, et a de grandes chances de détruire des paquets qui subissaient un détour, ce qui réduit le nombre de détours.

Toutefois, il est important de disposer de solutions pour garantir l'élimination de détours car :

- la purge de trames est appliquée arbitrairement sur les trames (dans les zones congestionnées, c'est-à-dire autour des nœuds impliqués dans des détours), indépendemment du

fait que ces trames provoquent des détours ou non,
- le risque de détours existe toujours au niveau routage, et la responsabilité de leur suppression ne peut pas incomber uniquement à la sous-couche MAC,
- la présence de détours engorge le réseau et donc est néfaste pour ses performances.

2.4 Suppression des détours

Quand plusieurs protocoles de routage coexistent dans un même réseau, des détours risquent d'apparaître. De tels détours doivent être évitées puisqu'elles augmentent les délais voire les pertes et surchargent le réseau. Dans cette partie, nous décrivons quelques protocoles de routage existants utilisés dans les réseaux de capteurs sans fil. Puis, nous proposons une approche en trois étapes pour permettre la coexistence de protocoles de routage quelconques. La première étape concerne les protocoles de routage compatibles, c'est-à-dire les protocoles qui peuvent coexister sans produire de détours dans le réseau. La deuxième étape concerne les protocoles de routage retardables, pour lesquels il est nécessaire d'utiliser une fonction de retard afin qu'ils soient compatibles. Cependant, cette fonction de retard peut être difficile à calculer pour des protocoles arbitraires. La troisième étape concerne les protocoles combinables. Cette étape fournit une fonction simple permettant de rendre compatibles des protocoles arbitraires.

2.4.1 Quelques protocoles de routage existants

Quelques protocoles de routage utilisés dans les réseaux de capteurs sans fil sont classifiés sur la figure 2.8. Ces protocoles sont répartis dans deux grandes catégories : les protocoles de routage réactifs et les protocoles de routage proactifs. Nous détaillons dans la suite les différents protocoles représentés sur la figure.

Les protocoles de routage réactifs, comme AODV [RFC 3561], exigent l'établissement d'un chemin de la source à la destination avant l'acheminement des paquets. Le temps nécessaire pour établir ce chemin peut être significatif puisqu'il dépend de la distance entre la source et la destination, et de la taille du réseau.

Afin de limiter le coût dans le réseau, AODV propose d'étendre la recherche progressivement. Initialement, la source diffuse une requête de route RREQ (pour *Route REQuest*) afin de connaître une route vers une certaine destination. La requête est diffusée pour un nombre de sauts limité. Si la source ne reçoit aucune réponse après un délai d'attente déterminé, elle retransmet une autre requête de route en augmentant le nombre maximum de sauts. Quand la

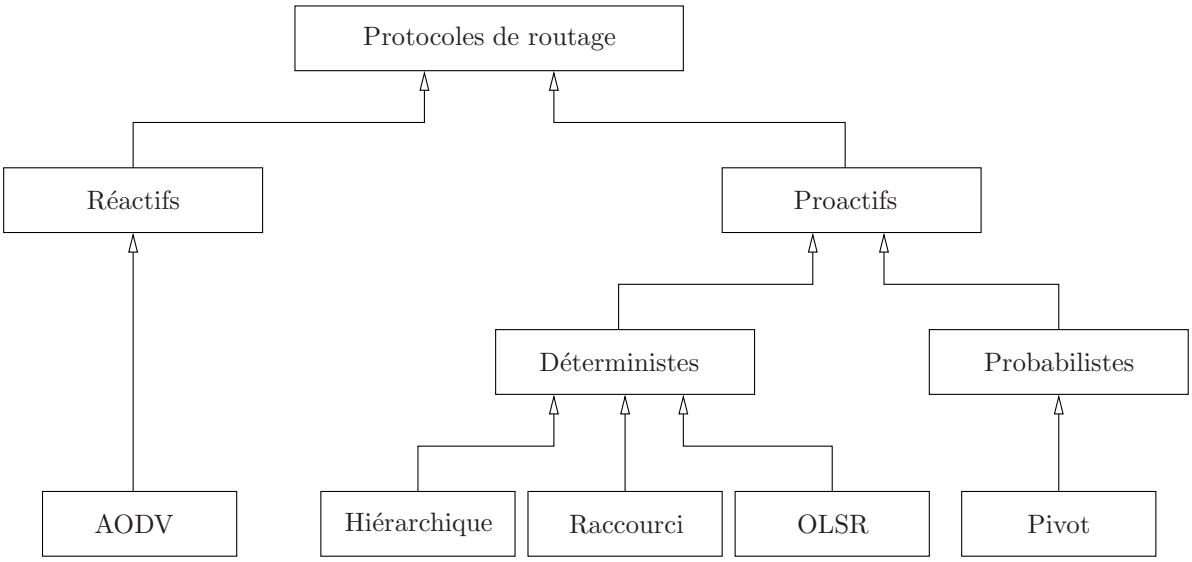


Figure 2.8 – Classification des protocoles de routage.

destination reçoit un message RREQ, elle renvoie un message RREP (pour *Route REPly*). Ce message est acheminé vers la source en suivant le même chemin traversé par RREQ. Des détails sur le routage effectué par AODV se trouve dans [RFC 3561].

Pour mesurer l'impact du temps nécessaire pour établir un chemin de la source vers la destination, nous avons considéré le scénario suivant : le réseau est constitué de 100 nœuds uniformément distribués dans une surface de 100 m $\times$ 100 m. La portée des nœuds est de20 m et le modèle de propagation utilisé est le *tow ray ground*. Nous sélectionnons le nœud 0 en tant que source, et le nœud 45, situé au centre du réseau, en tant que destination. Nous fixons le taux de transmission de paquets à un paquet par seconde.

La figure 2.9 illustre le délai de bout-en-bout moyen pour chaque paquet reçu correctement. Nous remarquons sur la figure que les paquets souffrent d'un grand délai. Ceci est dû au fait que la source diffuse un message de requête de chemin (RREQ pour *Route REQuest*) vers la destination afin d'établir un chemin optimal vers la destination. La découverte de route pénalise ainsi le délai de transmission des premiers paquets générés. Notons que les paquets de 1 jusqu'à 19 sont perdus[1] parce qu'AODV a été incapable d'établir un chemin entre la source et la destination. Le délai du premier paquet reçu (numéro 20) est de 9 secondes et celui du second paquet reçu est de 8 secondes. Pour chacun des premiers paquets reçus, le délai moyen diminue d'une seconde parce que les paquets sont transmis avec une fréquence d'un paquet par seconde. Après le neuvième paquet reçu, le temps d'établissement de route n'a plus d'impact sur le délai.

1. L'implémentation de cet exemple ne considère pas le tampon *(buffer)* : aucun paquet n'est stoqué dans le tampon avant ou bien durant l'établissement du chemin entre la source et la destination.

Le délai de bout-en-bout des paquets routés en utilisant les protocoles de routage réactifs est relativement grand. En effet, la durée de l'établissement des chemins pénalise ce délai.

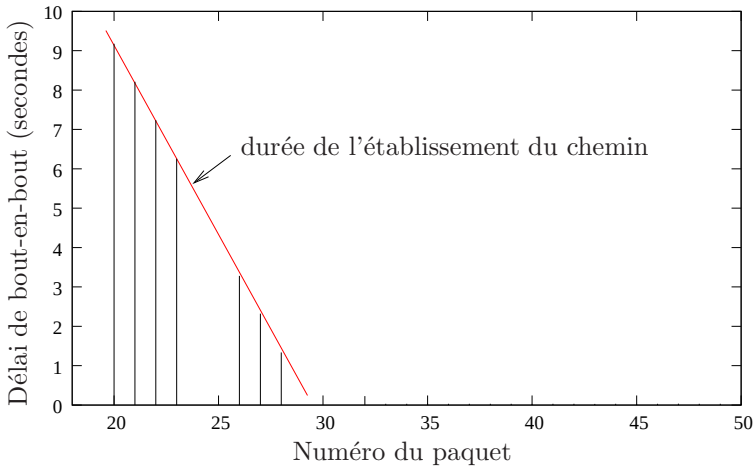


Figure 2.9 – Délai de bout-en-bout moyen avec AODV : dans cet exemple, 9 secondes sont nécessaires pour établir un chemin de la source vers la destination.

Les protocoles de routage proactifs sont capables de router les informations (même les plus urgentes) dès qu'elles sont produites car ils établissent les routes au préalable. Ils sont classés en deux sous-catégories : les protocoles déterministes et les protocoles probabilistes.

Avec les protocoles déterministes, les nœuds sources du réseau envoient des paquets à la destination en utilisant des chemins qui convergent rapidement. Néanmoins, les nœuds situés sur la partie commune des chemins doivent router les paquets générés par tous les nœuds sources. Cet état mène à une forte contention du médium quand les sources sont proches et qu'elles envoient à la même destination. Des exemples de protocoles proactifs déterministes incluent le protocole de routage hiérarchique, le protocole de routage raccourci [KKP+07] et OLSR [RFC 3626]. Les protocoles déterministes aboutissent donc à des congestions sur la plupart des chemins vers la destination.

Cependant, les protocoles de routage probabilistes se basent sur des nœuds pivots afin d'établir le chemin entre une source et une destination dans le réseau. Le choix des pivots dépend du réseau, de l'application et du trafic. Dans [LJK07], les auteurs proposent un protocole de routage efficace pour les réseaux de capteurs sans fil basé sur le routage avec des pivots. Le nœud source sélectionne un chemin et transmet les paquets à la destination *via* les nœuds pivots. Le nombre de messages de contrôle est réduit considérablement puisque les données sont transmises en suivant des nœuds pivots.

Dans la suite de cette partie, nous concentrons notre étude sur les protocoles de routage proactifs déterministe.

Protocole de routage raccourci

Le protocole de routage raccourci est proposé dans [KKP+07]. Il s'agit d'un protocole de routage proactif déterministe. Ce protocole améliore le protocole de routage hiérarchique en utilisant les tables de voisins. Si un nœud voisin peut réduire le nombre de sauts vers la destination par rapport au nœud calculé par le protocole de routage hiérarchique, ce nœud est choisi comme prochain saut. Pour cela, le nombre de sauts de chaque voisin vers la destination est calculé. Le nœud conduisant au plus petit nombre de sauts est sélectionné. Ce calcul est possible en connaissant uniquement l'adresse hiérarchique des voisins et les paramètres du réseau (L_m, R_m, et C_m).

La figure 2.10 montre un exemple de routage utilisant le protocole raccourci. Le nœud S souhaite envoyer un paquet au nœud destination D. Les nœuds voisins de S sont représentés par V_i. Le protocole de routage raccourci calcule pour chaque nœud V_i le nombre de sauts jusqu'à la destination en considérant que le chemin de V_i à D suit l'arbre. Dans l'exemple de la figure, le coût du chemin, affiché à côté de chaque nœud, peut être minimisé si l'émetteur transmet le paquet directement à son voisin V_3, car le chemin de V_3 à D sur l'arbre ne fait qu'un saut.

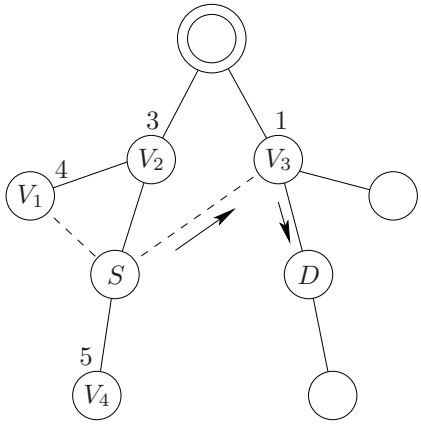


Figure 2.10 – Candidats pour le prochain saut selon le protocole de routage raccourci.

Protocoles de routage calculant le plus court chemin

Les protocoles de routage calculant le plus court chemin sont soit basés sur l'inondation soit basés sur l'algorithme de Dijkstra [Dij59]. Dans ce deuxième cas, connaissant la topologie globale du réseau, chaque nœud est capable de déterminer le plus court chemin de lui-même vers la destination, et donc d'acheminer les paquets. Ces protocoles trouvent généralement des chemins optimaux en terme de nombre de sauts.

Considérons l'exemple de la figure 2.11, et supposons que le nœud E souhaite envoyer un pa-

quet au nœud C. Le plus court chemin calculé est (E, D, C) et le nombre de sauts correspondant est égal à 2.

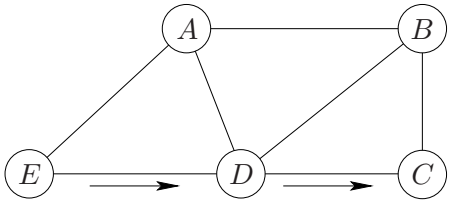


Figure 2.11 – Exemple de topologie. Le chemin de E à C est (E, D, C) avec un protocole calculant le plus court chemin.

Protocole de routage OLSR. Le protocole de routage OLSR *(Optimized Link-State Routing)* a été conçu pour les réseaux ad hoc mobiles de type MANET *(Mobile Ad hoc for NETworks)*. Il peut aussi être utilisé dans les réseaux de capteurs sans fil. OLSR est un protocole de routage proactif déterministe dans lequel les nœuds échangent périodiquement des informations concernant la topologie du réseau, dans le but d'établir un chemin vers n'importe quelle destination. Ce protocole est une adaptation optimisée pour réduire le nombre de messages de contrôle en utilisant le concept de nœuds relais multipoints MPR (pour *Multi-Point Relay*).

Un MPR d'un nœud n est un voisin à un saut de n, et qui est situé à portée de plusieurs nœuds qui sont voisins à deux sauts de n. Si n souhaite disséminer une information à plusieurs nœuds, n la transmet à tous ses MPR qui la retransmettent à leur tour (ce qui est plus efficace que si tous les voisins à un saut de n la retransmettaient). Les MPR permettent aussi de réduire la taille des messages de contrôle : au lieu de déclarer tous les liens dans le réseau, un nœud déclare seulement les liens avec ses voisins qui l'ont sélectionné comme MPR. Ainsi, seuls les MPR acheminent les messages de contrôle. Cette technique réduit donc significativement le nombre de transmissions des messages diffusés.

Lorsqu'un nœud n doit envoyer un paquet à un autre nœud d, n détermine si d est un voisin à un saut ou non. Si c'est le cas, n envoie directement le paquet à d. Sinon, n envoie le paquet à un MPR qui permet d'accéder à un voisin à deux sauts de n et qui est situé sur le plus court chemin de n à d. Ainsi, les routes suivies avec OLSR sont toujours des plus court chemins.

Pour expliquer plus en détail le fonctionnement des MPR tel qu'il est décrit dans [RFC 3626], nous utilisons la notation utilisée dans [MM09] pour représenter les voisins à un saut et à deux sauts. L'ensemble des MPR d'un nœud n est sélectionné selon un algorithme complexe. n calcule sa table de voisins à un saut $\mathcal{N}_1(n)$. Puis, chaque nœud v voisin de n construit sa propre table de voisins à un saut $\mathcal{N}_1(v)$, et l'envoie au nœud n. n est ainsi capable de calculer sa table de

voisins à deux sauts (exactement) selon la formule suivante : $\mathcal{N}_2(n) = \left(\cup_{v \in \mathcal{N}_1(n)} \mathcal{N}_1(v)\right) \setminus \mathcal{N}_1(n)$. Tous les nœuds de $\mathcal{N}_2(n)$ doivent être connectés *via* les MPR du nœud n. n détermine ses MPR en deux temps :

- Tout d'abord, n choisit comme MPR tous les nœuds de $\mathcal{N}_1(n)$ qui sont les seuls à connecter un nœud de $\mathcal{N}_2(n)$. En d'autres termes, s'il n'existe qu'un seul nœud m à un saut de n qui se trouve entre n et un voisin à deux sauts de n, ce nœud m est choisi comme MPR.
- Ensuite, n choisit successivement chaque MPR suivant en trouvant le voisin v qui maximise le nombre de nœuds restants à connecter dans $\mathcal{N}_2(n)$. n ajoute alors v à son ensemble de MPR. À présent, il ne reste plus à couvrir que $\mathcal{N}_2(n) \setminus \mathcal{N}_1(m)$. n cherche ensuite un autre MPR, jusqu'à ce que tous les nœuds de $\mathcal{N}_2(n)$ soient couverts, c'est-à-dire jusqu'à ce que $\cup_{m \in MPR(n)} \mathcal{N}_1(m) = \mathcal{N}_2(n)$. Ce choix n'est pas optimal, mais il s'agit d'une bonne heuristique pour le problème NP-complet de couverture d'ensemble.

Considérons l'exemple de la topologie illustrée dans la figure 2.12. Les lignes pleines représentent les liens de l'arbre alors que les lignes en pointillées représentent les liens de voisinage. Considérons que le nœud G souhaite envoyer un paquet au nœud F. OLSR, tout d'abord, détermine les MPR du nœud G. L'ensemble des voisins à exactement deux sauts de G est $\{B, C, F\}$. G sélectionne E comme MPR puisqu'il est le seul nœud qui connecte G à C en deux sauts. Une fois que le nœud E est choisi en tant que MPR, l'ensemble des nœuds voisins de G à deux sauts mais non connectés par les MPR se réduit à $\{\}$. Quand G doit envoyer un paquet à F, il choisit comme prochain saut le MPR qui assure le plus court chemin vers F. Dans ce cas, il s'agit du nœud E. Ensuite, E achemine le paquet à F. Avec cette topologie, le chemin de G à F est (G, E, F), et le nombre de sauts correspondant à ce chemin est 2. En utilisant le protocole de routage hiérarchique, le nombre de sauts est 5 puisque le paquet va suivre le chemin (G, D, B, A, C, F). Même avec le protocole de routage raccourci, le nombre de sauts est 4 parce que le paquet va suivre le chemin (G, D, B, C, F).

Le protocole EOLSR *(Energy efficient extension of OLSR)* [MM08a] est une extension d'OLSR économe en énergie. Ce protocole intègre la métrique de l'énergie résiduelle des nœuds dans le choix des MPR et dans l'algorithme de sélection des chemins de routage.

2.4.2 Protocoles de routage compatibles

Soit $\mathcal{R}$ un protocole de routage, et soit V un ensemble de nœuds. Pour chaque destination $d \in V$ et pour tout nœud $n \in V \setminus \{d\}$, le prochain saut de n, pour arriver à la destination d, en utilisant $\mathcal{R}$, est noté $\mathcal{R}(n, d)$. Nous supposerons que $\mathcal{R}(d, d)$ n'est pas défini.

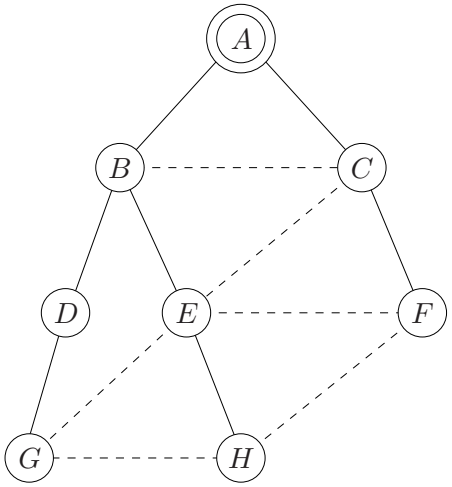


Figure 2.12 – Exemple de topologie. Le chemin de G à F est (G, D, B, A, C, F) avec le protocole hiérarchique, (G, D, B, C, F) avec le protocole raccourci et (G, E, F) avec OLSR.

Soit $\mathcal{R}_1$ et $\mathcal{R}_2$ deux protocoles de routage. Nous définissons la notion de détour par le fait que le chemin suivi par un paquet selon deux protocoles de routage $\mathcal{R}_1$ et $\mathcal{R}_2$ passe plus d'une fois par un même nœud.

Définition 1 (Protocoles de routage compatibles)**.** *Deux protocoles de routage $\mathcal{R}_1$ et $\mathcal{R}_2$ sont compatibles si tout nœud peut décider arbitrairement (en fonction de la sous-couche MAC à l'instant t) d'acheminer un paquet selon $\mathcal{R}_1$ ou bien selon $\mathcal{R}_2$, sans générer un détour, pour atteindre la destination d.*

La figure 2.13 présente un exemple avec deux protocoles de routage $\mathcal{R}_1$ et $\mathcal{R}_2$. Si un paquet est routé selon $\mathcal{R}_1$ de la source s jusqu'à m et selon $\mathcal{R}_2$ de m jusqu'à x, le même paquet passe deux fois par x avant d'arriver à la destination d. Ceci a lieu même si x et m ne sont pas des voisins. Dans ce cas, $\mathcal{R}_1$ et $\mathcal{R}_2$ ne sont pas compatibles.

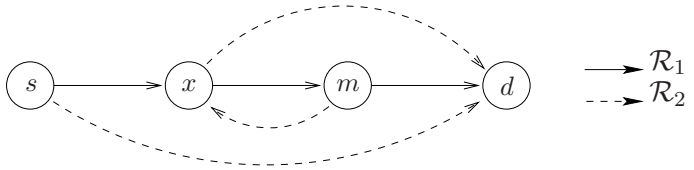


Figure 2.13 – Les deux protocoles de routage $\mathcal{R}_1$ and $\mathcal{R}_2$ ne sont pas compatibles.

La figure 2.14 illustre le même exemple de réseau avec deux autres protocoles de routage, aussi représentés par $\mathcal{R}_1$ et $\mathcal{R}_2$. Bien que $\mathcal{R}_1$ et $\mathcal{R}_2$ conduisent à des chemins différents, il n'est pas possible qu'un paquet fasse un détour. Dans ce cas, $\mathcal{R}_1$ et $\mathcal{R}_2$ sont compatibles.

Propriété 1. *Soit $\mathcal{R}_1$ et $\mathcal{R}_2$ deux protocoles de routage, soit V un ensemble de nœuds et $d \in V$*

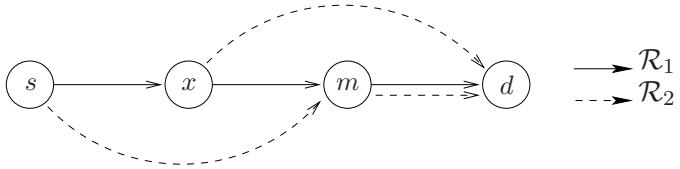


Figure 2.14 – Les deux protocoles de routage $\mathcal{R}_1$ et $\mathcal{R}_2$ sont compatibles, puisque chaque nœud peut déterminer indépendemment s'il faut router les paquets selon $\mathcal{R}_1$ ou bien selon $\mathcal{R}_2$.

une destination. S'il existe une fonction $f_d : V \to \mathbb{N}$ *telle que* $f_d(d) = 0$ *et* $\forall n \in V \backslash \{d\}$, $\max\{f_d(\mathcal{R}_1(n,d)), f_d(\mathcal{R}_2(n,d))\} < f_d(n)$*, alors* $\mathcal{R}_1$ *et* $\mathcal{R}_2$ *sont compatibles.*

Démonstration. Soit $d \in V$ une destination. Supposons qu'il existe une fonction f_d qui satisfasse la propriété $f_d(d) = 0$ et $\forall n \in V \backslash \{d\}$, $\max\{f_d(\mathcal{R}_1(n,d), f_d(\mathcal{R}_2(n,d))\} < f_d(n)$. Nous allons montrer que tout chemin partant d'un nœud arbitraire n atteint d en un nombre de sauts fini, sans générer de détour. En d'autres termes, nous allons montrer que pour toute suite (r_i) de décisions de routage (avec $r_i \in \{1, 2\}$ pour tout i), si (r_i) est suffisamment longue pour atteindre d (éventuellement elle peut être infinie si d n'est jamais atteint), l'unique chemin partant de n et suivant les décisions de routage (r_i) atteint d en un nombre de sauts fini, sans générer de détours.

Soit (r_i) une suite suffisamment longue de décisions de routage. Définissions le chemin suivant les décisions de routage (r_i) par $p = (n_0, n_1, n_2, \dots)$, avec $n_0 = n$ et $n_{i+1} = \mathcal{R}_{r_i}(n_i, d)$ pour tout nœud $n_i \in V \backslash \{d\}$. Par construction, p est unique (car (r_i) est fixée).

Montrons tout d'abord que :

– si p est infini, tous les nœuds de p appartiennent à $V \backslash \{d\}$,

– si p est fini, tous les nœuds de p, à l'exception du dernier, appartiennent à $V \backslash \{d\}$.

Si p est infini, à chaque nœud n_i de p correspond un nœud n_{i+1} dans p, ce qui n'est possible que si $n_i \in V \backslash \{d\}$. Ainsi, tous les nœuds de p sont dans $V \backslash \{d\}$. Si p est fini, on peut l'écrire sous la forme $(n_0, n_1, n_2, \dots, n_k)$. Par construction, pour tout $i \in [0; k-1]$, n_{i+1} existe, ce qui signifie que $n_i \in V \backslash \{d\}$. n_k n'est pas dans $V \backslash \{d\}$ car n_{k+1} n'est pas défini (notons que la suite (r_i) est choisie suffisamment longue).

Montrons que p est fini. Faisons un raisonnement par l'absurde en supposant que p est infini. Puisque p est infini, p ne contient que des nœuds de $V \backslash \{d\}$. À partir de p, on peut construire la suite infinie $s = (f_d(n_0), f_d(n_1), f_d(n_2), \dots)$ en appliquant la fonction f_d à chaque nœud de la suite p. Remarquons que pour tout n_i de p, puisque n_i appartient à $V \backslash \{d\}$, on a $f_d(n_{i+1}) = f_d(\mathcal{R}_{r_i}(n_i, d)) < f_d(n_i)$ par définition de f_d. La suite s est donc strictement décroissante. Or, s prend ses valeurs dans $\mathbb{N}$. Il n'est pas possible d'avoir une suite s infinie

strictement décroissante à valeurs dans $\mathbb{N}$: l'hypothèse que p était infini est donc fausse, et p est une suite finie.

À présent que nous avons montré que p est fini, montrons que p atteint d. Comme p est fini, tous les nœuds de p, à l'exception du dernier n_k, appartiennent à $V\backslash\{d\}$. Le nœud n_k est tel que $n_k \in V$ et n_k n'appartient pas à $V\backslash\{d\}$. n_k est donc égal à d (ce qui est la définition d'un protocole de routage). En d'autres termes, si n_k appartenait à $V\backslash\{d\}$, n_{k+1} existerait dans p, et n_k ne serait pas le dernier nœud de p.

À présent que nous savons que p est fini et que p finit en d, montrons que p est sans détour. Faisons un raisonnement par l'absurde en supposant que p passe deux fois par un même nœud. Il existe donc x et y tel que $n_x = n_y$, avec $x < y$. Construisons la suite $s = (f_d(n_0), f_d(n_1), f_d(n_2), \ldots, f_d(n_x), \ldots, f_d(n_y))$. Considérons à présent deux cas :

- y est égal à k. Cela signifie que $n_y = n_k$, or $n_x = n_y$ et $n_k = d$, donc $n_x = d$. Nous avons une contradiction car n_{x+1} existe dans p (car il existe un n_y avec y strictement supérieur à x) mais n'est pas défini car $n_x = d$. En d'autres termes, la contradiction vient du fait que la suite se termine en n_x.
- y n'est pas égal à k. Ainsi, n_y n'est pas le dernier nœud de p. En conséquence, tous les nœuds de p de n_0 à n_y appartiennent à $V\backslash\{d\}$. Ainsi, pour tout $n_i \in (n_0, n_1, n_2, \ldots, n_x, \ldots, n_y)$, on a $f_d(n_{i+1}) = f_d(\mathcal{R}_{r_i}(n_i, d)) < f_d(n_i)$. La suite s est donc strictement décroissante, ce qui impose que $f_d(n_x) < f_d(n_y)$ car $x < y$. Or, comme $n_x = n_y$, on a $f_d(n_x) = f_d(n_y)$. Nous avons ainsi une contradiction.

Dans les deux cas, nous arrivons à une contradiction. L'hypothèse que p passe deux fois par un même nœud est donc invalide.

Nous avons donc montré que l'unique chemin p atteint d en un nombre fini de sauts, sans générer de détour. $\mathcal{R}_1$ et $\mathcal{R}_2$ sont donc compatibles. □

De manière plus intuitive, la fonction f_d peut être vue comme une distance à la destination. Que le protocole de routage choisi soit $\mathcal{R}_1$ ou $\mathcal{R}_2$, le prochain nœud n_{i+1} correspond à une valeur $f_d(n_{i+1})$ plus petite que la valeur du nœud n_i qui est $f_d(n_i)$: dans tous les cas, le chemin p se rapproche de la destination. Pour tous les nœuds, à l'exception de d, il existe toujours un prochain saut selon $\mathcal{R}_1$ et $\mathcal{R}_2$: la construction du chemin s'arrête lorsque d est atteint. La convergence vers d est une conséquence de cette décroissance nécessaire en tout nœud (sauf en d).

Pour prouver que deux protocoles de routage donnés sont compatibles, la difficulté est de trouver la fonction f_d.

Théorème 1. *Le protocole de routage hiérarchique et le protocole de routage raccourci sont compatibles.*

Démonstration. Notons $\mathcal{R}_1$, le protocole de routage hiérarchique et $\mathcal{R}_2$, le protocole de routage raccourci. Construisons f_d de la manière suivante : pour tout nœud $n \in V$, $f_d(n)$ est le nombre de sauts restant à parcourir, sur l'arbre hiérarchique, pour atteindre la destination d.

Montrons que f_d respecte les conditions de la propriété 1. Tout d'abord, f_d est bien une fonction de V dans $\mathbb{N}$ et $f_d(d) = 0$. Pour tous les autres nœuds $n \in V \backslash \{d\}$:

- $f_d(n) > 0$ car $n \neq d$ et f_d est une distance.
- $f_d(n) > f_d(\mathcal{R}_1(n, d))$. En effet, comme $\mathcal{R}_1$ est le protocole de routage hiérarchique et que f_d est la distance selon l'arbre hiérarchique, le prochain saut de n selon $\mathcal{R}_1$ est à un saut de moins de la destination d que n. On a donc $f_d(n) = f_d(\mathcal{R}_1(n, d)) + 1$.
- $f_d(n) > f_d(\mathcal{R}_2(n, d))$. En effet, $\mathcal{R}_2$ est le protocole de routage raccourci. Posons $x = \mathcal{R}_2(n, d)$. x est le nœud parmi les voisins de n qui minimise la distance restant à parcourir pour atteindre d, c'est-à-dire que pour tout voisin v de n, $f_d(v) \geq f_d(x)$. Or, le prochain saut de n sur l'arbre hiérarchique, $\mathcal{R}_1(n, d)$, appartient aux voisins de n. On a donc $f_d(\mathcal{R}_1(n, d)) \geq f_d(x)$. En remarquant que $f_d(\mathcal{R}_1(n, d)) < f_d(n)$ (comme montré plus haut) et que $f_d(x) = \mathcal{R}_2(n, d)$, nous avons $f_d(n) > f_d(\mathcal{R}_2(n, d))$.

Donc, $f_d(n) > \max\{f_d(\mathcal{R}_1(n, d)), f_d(\mathcal{R}_2(n, d))\}$. □

Dans la suite, nous noterons d_t la distance d'un nœud à la destination sur l'arbre hiérarchique, et d_{sc} la distance d'un nœud à la destination en suivant le protocole de routage raccourci.

Théorème 2. *Le protocole de routage basé sur le plus court chemin et le protocole de routage OLSR sont compatibles.*

Démonstration. Notons $\mathcal{R}_1$, le protocole de routage basé sur le plus court chemin et $\mathcal{R}_2$, le protocole de routage OLSR. Construisons f_d de la manière suivante : pour tout nœud $n \in V$, $f_d(n)$ est le plus petit nombre de sauts à parcourir pour atteindre la destination d. Notons que $\mathcal{R}_1$ et $\mathcal{R}_2$ sont deux protocoles construisant des plus courts chemins.

Montrons que f_d respecte les conditions de la propriété 1. Tout d'abord, f_d est bien une fonction de V dans $\mathbb{N}$ et $f_d(d) = 0$. Pour tous les autres nœuds $n \in V \backslash \{d\}$:

- $f_d(n) > 0$ car $n \neq d$ et f_d est une distance.
- $f_d(n) > f_d(\mathcal{R}_1(n, d))$. Comme $\mathcal{R}_1$ est un protocole calculant le plus court chemin de n à d, le prochain saut $\mathcal{R}_1(n, d)$ est à un saut de moins de d que n. On a donc $f_d(n) = f_d(\mathcal{R}_1(n, d)) + 1$.

– $f_d(n) > f_d(\mathcal{R}_2(n,d))$. Cela se montre de la même manière que pour $\mathcal{R}_1$, car $\mathcal{R}_1$ et $\mathcal{R}_2$ sont deux protocoles calculant des plus courts chemins.

Donc, $f_d(n) > \max\{f_d(\mathcal{R}_1(n,d)), f_d(\mathcal{R}_2(n,d))\}$. □

Dans la suite, nous noterons d_{sp} la plus courte distance du nœud à la destination.

2.4.3 Protocoles de routage retardables

Toutes les combinaisons de protocoles de routage ne sont pas compatibles. Par exemple, le protocole de routage hiérarchique n'est pas compatible avec le protocole OLSR, et le protocole de routage raccourci n'est pas compatible avec un protocole de routage calculant le plus court chemin [2].

Quand deux protocoles de routage ne sont pas compatibles, nous proposons qu'un nœud puisse décider de conserver un paquet plutôt que de prendre le risque de générer un détour. L'acheminement d'un tel paquet est retardé jusqu'à ce qu'un autre protocole de routage puisse acheminer le paquet.

Définition 2 (Protocoles de routages retardables)**.** *Deux protocoles de routage $\mathcal{R}_1$ et $\mathcal{R}_2$ sont retardables par une fonction h_d, si tout paquet pour d atteint d sans entrer dans un détour et sans être conservé par un nœud pendant un temps infini. La conservation du paquet est basée sur le calcul suivant :*

– *si $h_d(n) \leq h_d(\mathcal{R}_{r_i}(n,d))$, le nœud n garde le paquet,*
– *sinon, le nœud n envoie le paquet selon $\mathcal{R}_{r_i}$, où $r_i \in \{1,2\}$ est déterminé localement par le nœud n. h_d est nommée la fonction de conservation.*

Propriété 2. *Soit $\mathcal{R}_1$ et $\mathcal{R}_2$ deux protocoles de routage, soit V un ensemble de nœuds et $d \in V$ une destination. S'il existe une fonction $h_d : V \to \mathbb{N}$ telle que $h_d(d) = 0$ et $\forall n \in V\backslash\{d\}$, $\min\{h_d(\mathcal{R}_1(n,d)), h_d(\mathcal{R}_2(n,d))\} < h_d(n)$, et si un même protocole de routage n'est pas considéré successivement pendant une infinité de tentatives de sauts, alors $\mathcal{R}_1$ et $\mathcal{R}_2$ sont retardables en utilisant la fonction h_d.*

Démonstration. Soit $d \in V$ une destination. Supposons qu'il existe une fonction h_d qui satisfasse la propriété $h_d(d) = 0$ et $\forall n \in V\backslash\{d\}$, $\min\{h_d(\mathcal{R}_1(n,d), h_d(\mathcal{R}_2(n,d))\} < h_d(n)$. Nous allons montrer que tout chemin partant d'un nœud arbitraire n atteint d en un nombre de sauts fini,

2. Cela pourrait être montré en exhibant un contre-exemple. Une conséquence de la non-compatibilité de ces deux paires de protocoles est montrée plus loin dans ce mémoire sur la figure 4.7 : la combinaison h-OLSR (correspondant au protocole de routage hiérarchique et au protocole OLSR) et la combinaison r-sp (correspondant au protocole de routage raccourci et au protocole calculant le plus court chemin) montrent des détours.

sans générer de détour (la conservation d'un paquet par un nœud n'étant pas considérée comme un détour). En d'autres termes, nous allons montrer que pour toute suite (r_i) de décisions de routage (avec $r_i \in \{1, 2\}$ pour tout i), si (r_i) est suffisamment longue, et si (r_i) ne contient pas une suite infinie de valeurs identiques consécutives, l'unique chemin partant de n et suivant les décisions (r_i) atteint d en un nombre de sauts fini, sans générer de détours.

Soit (r_i) une suite suffisamment longue de décisions de routage, ne contenant pas une suite infinie de valeurs identiques consécutives. Définissions le chemin suivant les décisions (r_i) par $p = (n_0, n_1, n_2, \dots)$, avec $n_0 = n$ et pour tout nœud $n_i \in V\backslash\{d\}$:

- si $h_d(n) \leq h_d(\mathcal{R}_{r_i}(n, d))$, alors n_i conserve le paquet, et donc $n_{i+1} = n_i$ (ce qui n'est pas un détour).
- sinon, $n_{i+1} = \mathcal{R}_{r_i}(n_i, d)$.

Par construction, p est unique (car (r_i) est fixée).

Montrons tout d'abord que :

- si p est infini, tous les nœuds de p appartiennent à $V\backslash\{d\}$,
- si p est fini, tous les nœuds de p, à l'exception du dernier, appartiennent à $V\backslash\{d\}$.

Si p est infini, à chaque nœud n_i de p correspond un nœud n_{i+1} dans p, ce qui n'est possible que si $n_i \in V\backslash\{d\}$. Ainsi, tous les nœuds de p sont dans $V\backslash\{d\}$. Si p est fini, on peut l'écrire sous la forme $(n_0, n_1, n_2, \dots, n_k)$. Par construction, pour tout $i \in [0; k-1]$, n_{i+1} existe, ce qui signifie que $n_i \in V\backslash\{d\}$. n_k n'est pas dans $V\backslash\{d\}$ car n_{k+1} n'est pas défini (notons que la suite (r_i) est choisie suffisamment longue).

Montrons que p est fini. Faisons un raisonnement par l'absurde en supposant que p est infini. Puisque p est infini, p ne contient que des nœuds de $V\backslash\{d\}$. À partir de p, on peut construire la suite infinie $s = (h_d(n_0), h_d(n_1), h_d(n_2), \dots)$ en appliquant la fonction h_d à chaque nœud de la suite p. Remarquons que pour tout n_i de p, puisque n_i appartient à $V\backslash\{d\}$, on a $h_d(n_{i+1}) \leq h_d(n_i)$, car :

- si $h_d(n_i) \leq h_d(\mathcal{R}_{r_i}(n_i, d))$, alors $h_d(n_{i+1}) = h_d(n_i)$ car $n_{i+1} = n_i$,
- sinon, nous avons $h_d(n_i) > h_d(\mathcal{R}_{r_i}(n_i, d))$, ce qui revient à dire que $h_d(n_{i+1}) = h_d(\mathcal{R}_{r_i}(n_i, d)) < h_d(n_i)$.

La suite s est donc décroissante (mais pas forcément strictement). Cependant, elle ne garde la même valeur que lorsque $h_d(n_i) \leq h_d(\mathcal{R}_{r_i}(n_i, d)$. Nous allons maintenant montrer que la suite s ne reste pas constante pendant un temps infini, ce qui revient à dire que si n_i conserve le paquet, il existe un $j = i + \alpha$, avec $\alpha > 0$ et α fini, tel que $h_d(n_j) > h(\mathcal{R}_{r_j}(n_j, d))$. Comme (r_i) ne contient pas une suite infinie de valeurs identiques consécutives, choisissons le plus

petit $j = i + \alpha$, avec $\alpha > 0$ et α fini, tel que $r_j \neq r_i$. Comme j est le plus petit entier valide, $n_i = n_{i+1} = \ldots = n_j$. Comme n_i a conservé le paquet, c'est que $h_d(n_i) \leq h_d(\mathcal{R}_{r_i}(n_i, d))$. Comme $n_i = n_j$, nous avons $h_d(n_j) \leq h_d(\mathcal{R}_{r_i}(n_j, d))$. Comme $r_j \neq r_i$, nous avons $\{r_i, r_j\} = \{1, 2\}$. De plus, nous savons que $h_d(n_j) > \min\{h_d(\mathcal{R}_1(n_j, d)), h_d(\mathcal{R}_2(n_j, d))\}$ par définition de h_d, ce qui signifie que $h_d(n_j) > h_d(\mathcal{R}_1(n_j, d))$ ou $h_d(n_j) > h_d(\mathcal{R}_2(n_j, d))$. Écrit autrement, nous pouvons dire que $h_d(n_j) > h_d(\mathcal{R}_{r_i}(n_j, d))$ ou $h_d(n_j) > h_d(\mathcal{R}_{r_j}(n_j, d))$. C'est obligatoirement la deuxième hypothèse qui est vraie, car $h_d(n_j) \leq h_d(\mathcal{R}_{r_i}(n_j, d))$ ce qu'il fallait obtenir. La suite s est donc décroissante mais ne reste jamais constante pendant un temps infini. Or, s prend ses valeurs dans $\mathbb{N}$. Il n'est pas possible d'avoir une suite s infinie décroissante, constante uniquement pendant un temps fini, et à valeurs dans $\mathbb{N}$: l'hypothèse que p était infini est donc fausse, et p est une suite finie.

À présent que p est fini, montrons que p atteint d. Comme p est fini, tous les nœuds de p, à l'exception du dernier n_k, appartiennent à $V \backslash \{d\}$. Le nœud n_k est tel que $n_k \in V$ et n_k n'appartient pas à $V \backslash \{d\}$. n_k est donc égal à d. En d'autres termes, si n_k appartenait à $V \backslash \{d\}$, n_{k+1} existerait dans p, et n_k ne serait pas le dernier nœud de p.

À présent que nous savons que p est fini, et que p finit en d, montrons que p est sans détour. Tout d'abord, remarquons que si il existe x tel que $n_x = n_{x+1}$ dans p (n_{x+1} étant le successeur immédiat de n_x), c'est que le paquet a été conservé par n_x. Par l'absurde, supposons que le paquet n'est pas conservé par n_x. On a alors $h_d(n_x) > h_d(\mathcal{R}_{r_x}(n_x, d)) = h_d(n_{x+1})$. Or, $n_x = n_{x+1}$, donc $h_d(n_x) = h_d(n_{x+1})$, ce qui est contradictoire. Nous avons donc montré que le seul moyen pour qu'un nœud n_x soit égal au nœud n_{x+1} dans p est que n_x conserve le paquet. Montrons à présent qu'un même paquet ne passe pas deux fois par un même nœud, hors conservation. Par l'absurde, comme on ne prend pas en compte les conservations, c'est donc qu'il existe $x < y < z$ tels que $n_x \neq n_y$ (car n_x n'a pas conservé) et tel que $n_x = n_z$. Construisons la suite $s = (h_d(n_0), h_d(n_1), h_d(n_2), \ldots, h_d(n_x), \ldots, h_d(n_y), \ldots, h_d(n_z))$. Considérons deux cas :

- Si $n_z = n_k$, c'est que $n_z = d$. Ainsi, $n_x = d$, et le successeur de x n'est pas défini. Le chemin p ne peut pas passer par n_y, ce qui est une contradiction.
- Si $n_z \neq n_k$, c'est donc que p ne termine pas en n_z, et ainsi pour tout nœud de p de n_0 à n_z appartiennent à $V \backslash \{d\}$. Ainsi, pour tout $n_i \in (n_0, n_1, n_2, \ldots, n_x, \ldots, n_y, \ldots, n_z)$, on a $h_d(n_{i+1}) \leq h_d(\mathcal{R}_{r_i}(n_i, d)) \leq h_d(n_i)$. De plus, comme n_x n'a pas conservé le paquet, nous avons $h_d(n_{x+1}) < h_d(n_x)$, ce qui signifie que $h_d(n_y) < h_d(n_x)$. De la même manière, comme $n_y \neq n_z$ (car $n_y \neq n_x$ et $n_x = n_z$), $h_d(n_z) < h_d(n_y)$. Nous avons donc $h_d(n_z) < h_d(n_x)$. Or, $n_z = n_x$ donc $h_d(n_z) = h_d(n_x)$. Nous avons donc une contradiction.

Dans les deux cas, nous arrivons à une contradiction. L'hypothèse que p passe deux fois par un même nœud est donc invalide.

Nous avons donc montré que l'unique chemin p atteint d en un nombre fini de sauts, sans générer de détour. $\mathcal{R}_1$ et $\mathcal{R}_2$ sont donc retardables par h_d. □

Théorème 3. *Le protocole de routage hiérarchique et le protocole de routage OLSR sont retardables si la distance sur l'arbre d_t (respectivement la distance du plus court chemin d_{sp}), est utilisée comme fonction de conservation.*

Démonstration. Ceci est conclu de la propriété 2 en utilisant la distance sur l'arbre d_t (respectivement la distance du plus court chemin d_{sp}), comme fonction de conservation. En effet, pour un nœud n du réseau, la distance minimale de son prochain saut vers la destination en suivant le protocole hiérarchique (respectivement en suivant le protocole OLSR) que nous nommons $\mathcal{R}_1$ est toujours inférieure à la distance de n vers la destination en utilisant d_t (respectivement en utilisant d_{sp}). Nous avons donc $h_d(n) > h_d(\mathcal{R}_1(n,d))$ et donc $h_d(n) > \min\{h_d(\mathcal{R}_1(n,d), h_d(\mathcal{R}_2(n,d))\}$. □

Théorème 4. *Le protocole de routage raccourci et tout protocole de routage calculant le plus court chemin sont retardables si la distance du plus court chemin d_{sp} (respectivement la distance sur l'arbre d_t), est utilisée comme fonction de conservation.*

Démonstration. Ceci est conclu de la propriété 2 en utilisant d_t (respectivement d_{sp}) comme fonction de conservation. La même preuve que celle du théorème 3 s'applique sur ce théorème. □

L'approche basée sur les protocoles retardables possède deux inconvénients principaux. Premièrement, un paquet peut être gardé par un nœud pour une longue durée. Deuxièmement, il peut être difficile de trouver (ou de calculer au niveau de la sous-couche MAC) une fonction de conservation pour des protocoles complexes.

2.4.4 Protocoles de routage combinables

L'approche des protocoles de routage combinables consiste à utiliser un protocole particulier $\mathcal{R}^*$ (dont la fonction de distance est connue) en supplément à des protocoles de routage existants $\mathcal{R}_1$ et $\mathcal{R}_2$. Alors qu'il peut être difficile de trouver une fonction de conservation pour rendre les protocoles $\mathcal{R}_1$ et $\mathcal{R}_2$ retardables, la tâche devient plus facile pour les protocoles $\mathcal{R}_1$ et $\mathcal{R}_2 \cup \mathcal{R}^*$,

$\mathcal{R}_1 \cup \mathcal{R}^*$ et $\mathcal{R}_2$, ou $\mathcal{R}_1 \cup \mathcal{R}^*$ et $\mathcal{R}_2 \cup \mathcal{R}^*$, $\cup$ représentent la juxtaposition de deux protocoles de routage.

La superposition d'un protocole de routage primaire $\mathcal{R}_i$ avec un protocole de routage secondaire $\mathcal{R}^*$ fonctionne comme suit. Soit V un ensemble de nœuds, soit $d \in V$ une destination et soit $f_d^* : V \to I\!N$ une fonction telle que $f_d^*(d) = 0$ et $\forall n \in V \backslash \{d\}$, $f_d^*(\mathcal{R}^*(n,d)) < f_d^*(n)$. f_d^* est considérée connue puisque $\mathcal{R}^*$ n'est pas arbitraire. Pour router un paquet à la destination d, un nœud n détermine si $f_d^*(\mathcal{R}_i(n,d)) < f_d^*(n)$ ou non. Si c'est le cas, le paquet est envoyé selon le protocole de routage $\mathcal{R}_i$. Sinon, le paquet est envoyé selon le protocole de routage secondaire $\mathcal{R}^*$.

Propriété 3. *Considérons trois protocoles de routage $\mathcal{R}_1$, $\mathcal{R}_2$ et $\mathcal{R}^*$, tels que un même protocole de routage ($\mathcal{R}_1$ ou $\mathcal{R}_2 \cup \mathcal{R}^*$) n'est pas considéré successivement pendant une infinité de tentative de sauts. Considérons un ensemble de nœuds V et une destination $d \in V$. Soit $f_d^* : V \to I\!N$ telle que $f_d^*(d) = 0$ et $\forall n \in V \backslash \{d\}$, $f_d^*(\mathcal{R}^*(n,d)) < f_d^*(n)$. $\mathcal{R}_1$ et $\mathcal{R}_2 \cup \mathcal{R}^*$ sont retardables en utilisant la fonction f_d^*.*

Démonstration. Nous avons seulement chercher à montrer que $\forall n \in V \backslash \{d\}$, $\min\{f_d^*(\mathcal{R}_1(n,d)), f_d^*((\mathcal{R}_2 \cup \mathcal{R}^*)(n,d))\} < f_d^*(n)$. Pour router un paquet jusqu'à la destination d selon $\mathcal{R}_2 \cup \mathcal{R}^*$, un nœud n détermine si $f_d^*(\mathcal{R}_2(n,d)) < f_d^*(n)$ ou non. Si $f_d^*(\mathcal{R}_2(n,d)) < f_d^*(n)$, n route le paquet selon $\mathcal{R}_2(n,d)$. Dans ce cas, $f_d^*((\mathcal{R}_2 \cup \mathcal{R}^*)(n,d)) = f_d^*(\mathcal{R}_2(n,d)) < f_d^*(n)$. Si $f_d^*(\mathcal{R}_2(n,d)) \geq f_d^*(n)$, n route le paquet selon $\mathcal{R}^*$. Dans ce cas, $f_d^*((\mathcal{R}_2 \cup \mathcal{R}^*)(n,d)) = f_d^*(\mathcal{R}^*(n,d)) < f_d^*(n)$, par définition de f_d^*. Dans les deux cas, nous avons $f_d^*((\mathcal{R}_2 \cup \mathcal{R}^*)(n,d)) < f_d^*(n)$. Donc, $\min\{f_d^*(\mathcal{R}_1(n,d)), f_d^*((\mathcal{R}_2 \cup \mathcal{R}^*)(n,d))\} < f_d^*(n)$, ce qui complète la preuve. □

L'approche des protocoles de routage combinables utilise un protocole de routage connu $\mathcal{R}^*$ pour assurer que deux protocoles de routage arbitraires soient retardables. $\mathcal{R}^*$ est choisi tel que (i) f_d^* est connue et facile à calculer, (ii) la surcharge de $\mathcal{R}^*$ (en termes de paquets de contrôles et d'énergie) est raisonnable par rapport à celle des autres protocoles. Nous proposons d'utiliser le protocole de routage hiérarchique pour $\mathcal{R}^*$, à chaque fois qu'une topologie en arbre est utilisée par l'un des protocoles primaires. Dans ce cas, la surcharge due à la maintenance de l'arbre est partagée par le protocole $\mathcal{R}^*$ et par le protocole primaire.

La figure 2.15 représente l'ordonnancement de notre approche de protocoles combinables. Notons que le protocole $\mathcal{R}^*$ est utilisé seulement quand un nœud devrait garder un paquet.

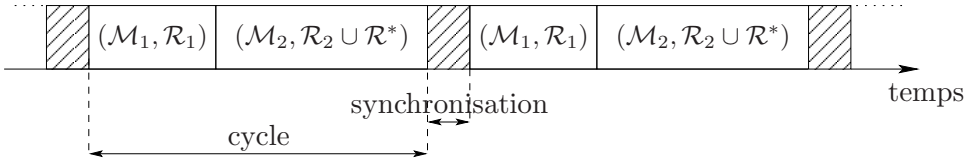


Figure 2.15 – Un exemple d'ordonnancement de protocoles combinables, où $\mathcal{R}^*$ est combiné avec $\mathcal{R}_2$. Notons que $\mathcal{R}^*$ pourrait aussi être combiné avec $\mathcal{R}_1$ seulement, ou bien avec les deux protocoles $\mathcal{R}_1$ et $\mathcal{R}_2$.

$\mathcal{R}_1 \backslash \mathcal{R}_2$	**hiérarchique**	**raccourci**	**OLSR**	**plus court chemin**
hiérarchique	compatible	compatible	retardable (avec d_t ou d_{sp})	retardable (avec d_t ou d_{sp})
raccourci	compatible	compatible	retardable (avec d_t, d_{sc} ou d_{sp})	retardable (avec d_t, d_{sc} ou d_{sp})
OLSR	retardable (avec d_t ou d_{sp})	retardable (avec d_t, d_{sc} ou d_{sp})	compatible	compatible
plus court chemin	retardable (avec d_t ou d_{sp})	retardable (avec d_t, d_{sc} ou d_{sp})	compatible	compatible

Tableau 2.1 – Synthèse de l'ordonnancement des protocoles de routage.

2.4.5 Synthèse

Le tableau 2.1 récapitule les différents scénarios que nous avons étudiés dans cette section. Nous avons montré d'une manière théorique (et nous vérifions par des simulations détaillées plus loin dans cette thèse) que le protocole hiérarchique et le protocole raccourci sont compatibles, et qu'il en est de même pour les protocoles calculant les plus courts chemins. Le protocole de routage hiérarchique et les protocoles de routage calculant les plus courts chemins sont retardables quand la fonction de conservation est la distance sur l'arbre d_t ou bien quand il s'agit de la distance représentant le plus court chemin d_{sp}. Le protocole de routage raccourci et les protocoles de routage calculant le plus court chemin sont retardables quand la fonction de conservation est la distance du chemin du raccourci d_{sc}, ou bien la distance du plus court chemin d_{sp}, ou bien la distance sur l'arbre d_t (ceci parce que d_t est toujours plus grande ou bien égale à d_{sc}). Finalement, tous les protocoles étudiés sont combinables.

2.5 Conclusion

L'architecture basée sur l'utilisation d'un seul protocole de routage avec une seule méthode d'accès ne peut pas répondre à tous les besoins de QoS des applications. Ce chapitre a présenté une nouvelle architecture avec plusieurs protocoles par couche où plusieurs combinaisons de protocoles MAC $\mathcal{M}_i$ et réseau $\mathcal{R}_i$ sont activées à tour de rôle. Ce chapitre a détaillé les contraintes des différents protocoles MAC et routage utilisés dans notre architecture.

Cette architecture pose des problèmes de dimensionnement des périodes pour chaque application et augmente le délai de bout-en-bout. Un mécanisme de *cross-layering* a été proposé afin de résoudre ces problèmes. Ce mécanisme permet les échanges des paquets entre les différentes files d'attente. Un paquet marqué par une application $\mathcal{A}_i$ peut être envoyé par un couple $(\mathcal{M}_j, \mathcal{R}_j)$ si la file d'attente $\mathcal{Q}_j$ est vide. Cependant, ce mécanisme peut faire apparaître des détours dans le réseau quand les protocoles de routage changent pour l'envoi d'un même paquet. Nous avons proposé trois catégories de protocoles, et nous en avons déduit trois mécanismes qui permettent d'éliminer les détours dans le réseau afin de pouvoir appliquer les échanges de files d'attente.

Chapitre 3

Échanges des files d'attente dans MaCARI

La pile de protocoles proposée dans le cadre du projet OCARI a pour but de répondre aux problématiques des réseaux de capteurs sans fil industriels, à savoir une grande autonomie énergétique, des conditions de propagation radio difficiles, et des exigences multiples de QoS.

Dans ce chapitre, nous proposons d'implémenter un mécanisme de *cross-layering* permettant d'échanger les paquets entre les files d'attente existantes dans un réseau OCARI afin d'optimiser ses performances.

3.1 Introduction

Dans de nombreuses industries, il y a un fort besoin de surveiller les installations ou de garantir la sécurité du personnel. Les réseaux de capteurs sans fil sont bien adaptés à ces demandes et ont un avantage supplémentaire : ils ne sont pas coûteux à déployer. En revanche, assurer un fonctionnement du réseau pendant plusieurs années sans intervention pour changer les batteries est un défi important. En outre, la présence d'équipements métalliques dégrade beaucoup les conditions de propagation des signaux radio. C'est pourquoi des solutions spécifiques doivent être conçues.

La pile protocolaire OCARI est destinée à des applications industrielles. L'objectif principal des protocoles d'OCARI est le déterminisme et l'économie en énergie [MM08b].

Dans ce qui suit, nous détaillons le projet OCARI avec ses différents protocoles, et nous décrivons notre approche d'échanges de files d'attente, faisant intervenir la couche réseau et la sous-couche MAC. Ce travail s'appuie sur la pile protocolaire d'OCARI telle qu'elle était à la

fin du projet ANR OCARI en juin 2010.

3.2 Description du projet OCARI

Le projet OCARI (pour Optimisation des Communications Ad hoc pour les Réseaux Industriels) est basé sur un partenariat entre académiques et industriels et subventionné par l'ANR (Agence Nationale de Recherche) [ANR]. Ce projet a pour but de proposer une nouvelle génération de réseaux sans fil permettant la surveillance d'installations et d'édifices industriels. Il s'agit de concevoir des réseaux sans fil robustes prenant en compte les environnements fortement contraints et permettant de répondre aux besoins des applications industrielles.

3.2.1 Objectifs

Le projet OCARI vise à répondre à des applications variées ayant des contraintes de plusieurs types : un faible délai de bout-en-bout pour un type de trafic, une faible consommation d'énergie pour les nœuds alimentés par des batteries, le déterminisme et la fiabilité des communications.

Un réseau OCARI est composé d'îlots séparés (soit géographiquement, soit *via* des canaux différents) et pouvant chacun contenir quelques dizaines voire centaines de nœuds. Chacun de ces îlots possède une structure arborescente et une passerelle permettant son interconnexion éventuelle à une infrastructure (filaire ou non). Dans chaque îlot, une entité spécifique joue le rôle d'une passerelle entre le réseau formé dans l'îlot et l'infrastructure. Cette passerelle a pour but d'harmoniser les informations récoltées du réseau de capteurs, de transmettre ces informations aux applications, de surveiller et de mettre à jour les paramètres de configuration de ce réseau. La connectivité des îlots peut être assurer aussi grâce à une entité mobile appelée, rondier. Outre son rôle applicatif, le rondier est également utilisé pour véhiculer des informations entre les îlots.

La figure 3.1 présente un exemple d'un réseau OCARI avec trois îlots interconnectés par le biais de passerelles reliées à une unité de contrôle. Le rôle de cette unité de contrôle est de surveiller l'activité industrielle et mettre à jour les paramètres du réseau. Nous remarquons aussi un rondier qui se déplace entre les ilôts et qui se connecte de manière ponctuelle à l'un de ces îlots afin d'effectuer des interventions localisées ou de collecter des informations.

Dans le projet OCARI, deux types de trafic ont été identifiés :

– Des données générées plutôt périodiquement et destinées à être transmises au plus tôt à une unité de supervision. Ces données sont nommées plus tard par le trafic non contraint.

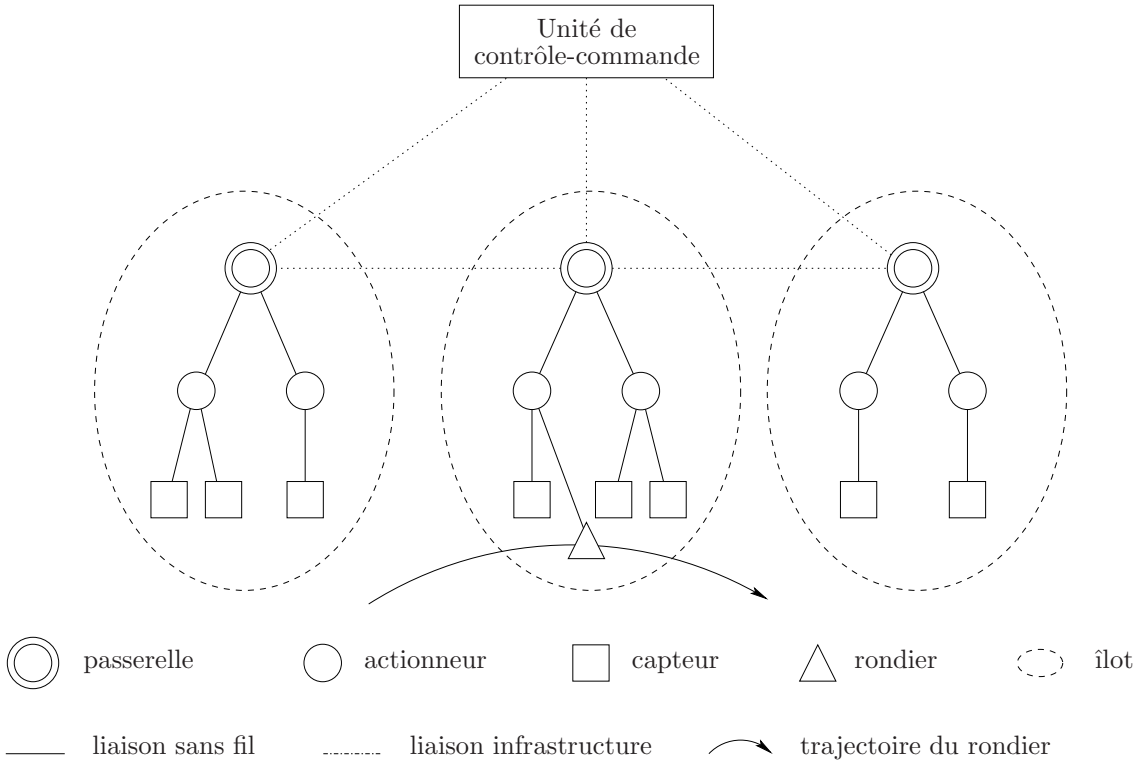


Figure 3.1 – Topologie du réseau industriel OCARI.

– Des données dont le délai de transit (la latence) dans le réseau doit être borné et le taux de pertes doit être très faible.

3.2.2 Partenaires

Le projet OCARI regroupe quatre partenaires académiques et trois partenaires industriels. Les partenaires académiques sont l'INRIA (Institut National de Recherche en Informatique et Automatique), le LATTIS (Laboratoire Toulousain de Technologie et d'Ingénierie des Systèmes), le LIMOS (Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes) et le LRI (Laboratoire de Recherche en Informatique). Les partenaires industriels sont l'ÉDF R&D (Électricité de France, Recherche et Développement), la DCNS (Direction de Construction Navale Sextant) et Télit. ÉDF R&D compte équiper ses sites de production d'énergie (tels que les barrages hydrauliques et les centrales nucléaires) de réseaux de capteurs. La DCNS a pour volonté de développer une technologie réseau à bord de navires de guerre, qui constitue des environnements fortement contraints. Enfin, Télit propose des prototypes de cartes et un savoir-faire au niveau de leur programmation.

La figure 3.2 montre la répartition des rôles entre les différents acteurs du projet, selon leur contribution sur la pile OCARI. Le travail de l'équipe Réseaux et Protocoles du LIMOS a été

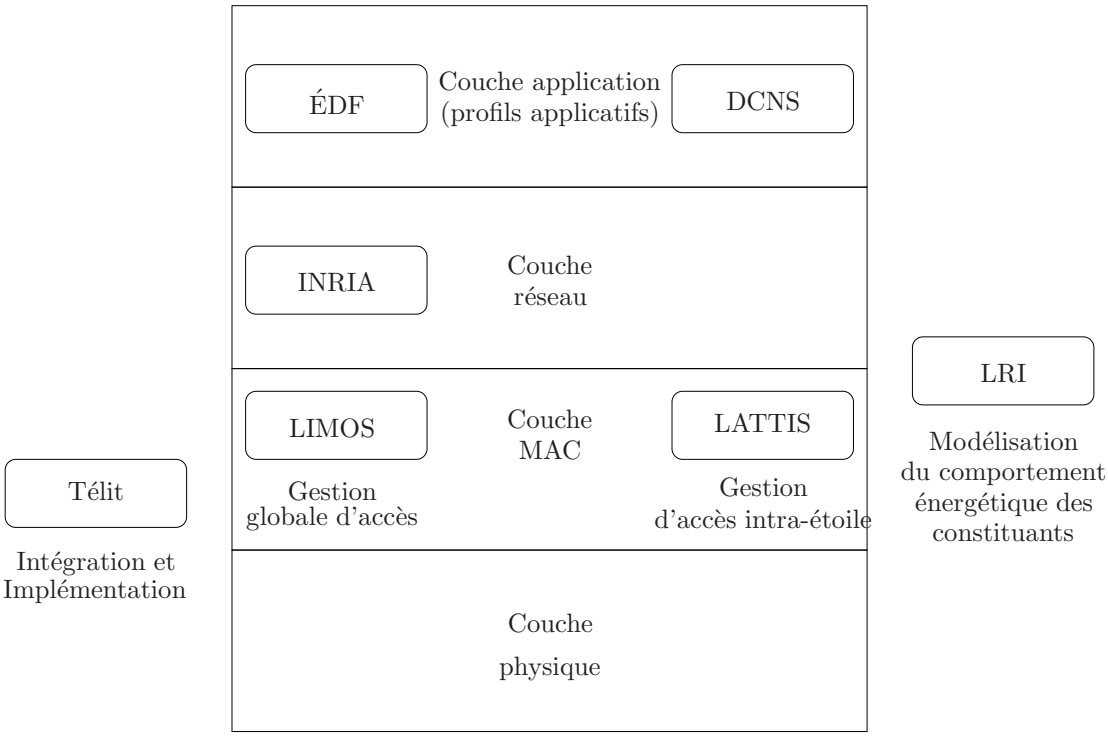


Figure 3.2 – Domaine d'intervention des différents acteurs sur la pile OCARI.

principalement axé sur l'élaboration d'un protocole d'accès au médium nommé MaCARI, et sur l'évaluation par simulation et par maquettage de ce protocole. Ce travail a été réalisé en permettant l'intégration des protocoles et des mécanismes développés par les autres partenaires grâce à la spécification d'interface et de primitives associées.

3.2.3 MaCARI : la méthode d'accès de la pile OCARI

MaCARI (pour MAC pour OCARI) [CLG+09] est une méthode d'accès au médium destinée aux réseaux de capteurs sans fil. Cette méthode d'accès fournit à la fois un certain déterminisme et permet l'économie d'énergie de tous les types de nœuds. Le déterminisme permet de garantir l'accès au médium. L'économie d'énergie consiste à optimiser l'activité de chaque entité du réseau pour faire en sorte qu'elle soit en état de sommeil le plus souvent possible, afin de prolonger au maximum l'autonomie de l'entité et par conséquent celle du réseau. MaCARI est un protocole MAC basé sur le standard IEEE 802.15.4 ; il repose sur la couche PHY de ce standard et propose des variantes à sa sous-couche MAC. Il reprend le mode d'adressage et le routage hiérarchique de ZigBee.

Description

MaCARI propose une différenciation de trafic pour un trafic associé à un délai de transit borné (appelé trafic contraint) et un trafic dont le délai de transit est non borné dans le temps (appelé trafic non-contraint). Une métaphore souvent utilisée explique les principes du protocole : on peut assimiler les communications dans MaCARI à la remontée des rivières par les saumons. Lorsqu'ils remontent la rivière, les saumons sont parfois confrontés à des obstacles comme des cascades. Pour faciliter le franchissement d'obstacles élevés, on peut mettre en place des « échelles à saumons », qui permettent aux saumons de poursuivre leur remontée au prix d'un délai plus élevé. Ainsi, les saumons ont deux options : essayer de remonter la cascade, ce qui n'est pas garanti mais éventuellement rapide ou emprunter l'échelle, ce qui est garanti mais lent. Dans MaCARI, une information peut être envoyée de deux manières : soit par une méthode qui ne garantit pas de délai borné jusqu'à sa destination, soit par une méthode qui garantit que la destination recevra le message dans un délai borné.

MaCARI a été conçu pour permettre :

- l'économie d'énergie au niveau deux de tous les éléments du réseau,
- la garantie d'un délai de bout-en-bout borné pour un certain trafic,
- l'étanchéité du réseau (c'est-à-dire qu'il n'y a pas d'interférence entre deux ilôts),
- une différenciation de service au niveau MAC.

MaCARI définit trois types de nœuds : le coordinateur du PAN (appelé CPAN), les coordinateurs, et les feuilles. Le CPAN est le nœud qui crée le réseau, établit la base d'une topologie arborescente, et maintient une synchronisation entre l'ensemble des nœuds du réseau. Il est souvent le puits de données ou la passerelle pour l'interconnexion à d'autres ilôts. Les coordinateurs routent les informations et participent à la phase de synchronisation du réseau. Un coordinateur peut néanmoins supporter des capteurs ou des actionneurs. Les feuilles (capteurs ou actionneurs) sont chacune attachées à un coordinateur et communiquent uniquement avec lui. Un coordinateur et ses feuilles constituent une étoile. Un exemple de topologie de réseau OCARI est illustré dans la figure 3.3.

Découpage temporel et accès au médium

MaCARI divise le temps en quatre périodes principales qui forment un cycle global. Ces périodes sont les suivantes.

- Une période de synchronisation $[T_0; T_1]$. Dans cette période, le CPAN diffuse une balise de synchronisation qui précise le dimensionnement du cycle global. Pour pouvoir atteindre

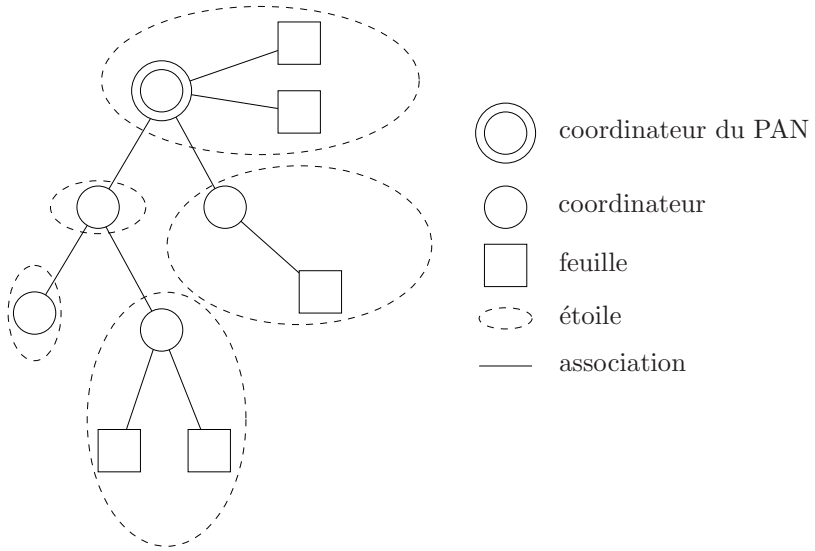


Figure 3.3 – Exemple de topologie d'un ilôt OCARI.

toutes les entités du réseau qui ne sont pas forcément à portée du CPAN, les coordinateurs propagent à tour de rôle, la balise diffusée par le CPAN. L'ordre dans lequel les coordinateurs doivent propager la balise est porté dans cette dernière.

– Une période d'activité ordonnancée $[T_1; T_2]$. Cette période est une période de séquencement et de relais. Dans cette période, à chaque étoile est allouée un intervalle de temps durant lequel le coordinateur de l'étoile communique avec les feuilles qui lui sont associées. Cet intervalle est appelé la période d'activité d'une étoile, et permet à chaque étoile d'organiser son activité sans craindre des collisions ou d'interférences dues à l'activité d'autres étoiles en un même lieu. Ainsi, un coordinateur est capable d'allouer un GTS (contribution du LATTIS dans OCARI) sans craindre que les intervalles de temps soient utilisés par le coordinateur d'une autre étoile. Un autre intervalle de temps supplémentaire est aussi réservé à chaque coordinateur (sauf au CPAN), durant lequel il communique avec son père. Cet intervalle de temps est appelé l'intervalle de relais. Ces intervalles de relais sont prévus pour les échanges de trames protégées qui doivent être envoyées sans risque de collisions. Ce sont des intervalles de temps à accès garanti étant donné que le coordinateur et son père sont les deux seules entités actives dans leur îlot, durant cet intervalle de temps. Ce découpage temporel est représenté dans la figure 3.4.
– Une période d'activité non-ordonnancée $[T_2; T_3]$. À l'instant T_2 toutes les étoiles ont eu une période d'activité qui a permis aux coordinateurs de récolter le trafic généré par leurs feuilles (ou confié à un actionneur) et relayer la partie contrainte de ce trafic à leurs pères ou leurs fils. L'acheminement du trafic non-contraint se fait dans la période $[T_2; T_3]$. Seuls

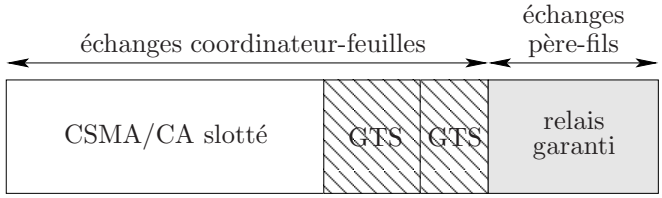


Figure 3.4 – Période d'activité d'une étoile suivie par un intervalle de relais.

les coordinateurs sont actifs durant cette période.

– une période d'inactivité $[T_3; T_0]$. Cette période correspond à une période durant laquelle tous les nœuds économisent de l'énergie en mettant leur module radio en mode sommeil. La durée de cette période dépend de la réactivité exigée par l'application. Plus cette période est grande, plus la consommation d'énergie est économisée mais plus le délai de transit à travers le réseau est grand.

Ces périodes sont présentées sur la figure 3.5.

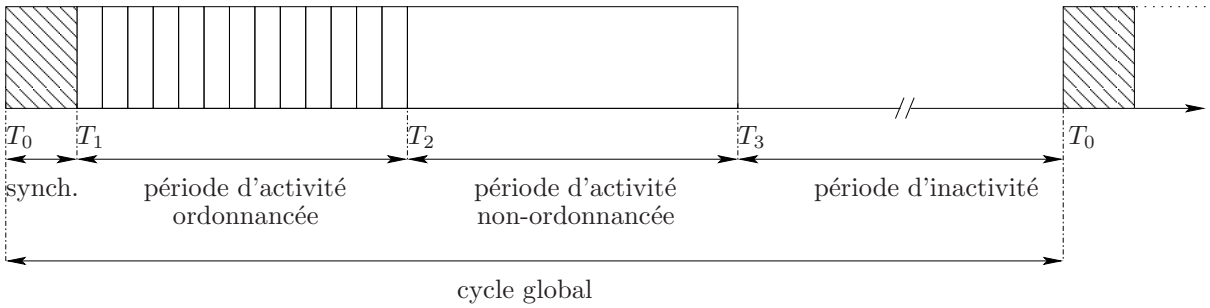


Figure 3.5 – Cycle global de MaCARI.

L'activité d'un nœud dépend de son type. Un coordinateur est actif durant la période $[T_0; T_1]$, de l'instant T_0 jusqu'à ce qu'il envoie sa balise, alors qu'une feuille est active de l'instant T_0 jusqu'à ce qu'elle reçoive une balise. Dans $[T_1; T_2]$, un coordinateur est actif durant la période d'activité de son étoile, l'intervalle de relais avec son père et durant les intervalles de relais de ses fils. Une feuille est active durant la période d'activité de son étoile seulement. Dans $[T_2; T_3]$, tous les coordinateurs sont actifs pour acheminer les paquets à leur destination finale, les feuilles ne sont pas actives. Finalement, aucune entités n'est active durant $[T_3; T_0]$.

L'accès au médium diffère dans MaCARI selon les intervalles de temps. Les balises dans $[T_0; T_1]$ sont envoyées directement sans CSMA/CA. Les échanges intra-étoile sont gérés en CSMA/CA slotté pour le trafic non-contraint et sans CSMA/CA durant les GTS pour le trafic contraint. Les échanges durant les intervalles de relais garanti sont effectués sans CSMA/CA. Durant la période $[T_2; T_3]$, l'accès est géré selon le protocole CSMA/CA slotté.

Synchronisation $[T_0; T_1]$

Le but de la synchronisation dans MaCARI est d'informer toutes les entités du réseau du découpage temporel du cycle global. Si ce découpage peut être conservé pour n cycles, la synchronisation ne peut être faite qu'une fois tous les n cycles. Toutefois, la synchronisation est généralement plus fréquente : il faut s'assurer que les entités ne se désynchronisent pas lorsque la fréquence de synchronisation est réduite et les changements de topologie ne peuvent être pris en compte qu'après la synchronisation.

Segmentation $[T_1; T_3]$

MaCARI utilise une segmentation à deux niveaux : une segmentation des activités des étoiles, et une segmentation pour garantir le relais d'un trafic contraint temporellement.

Séquencement des activités des étoiles. Durant la partie de la période d'activité qui lui est réservé, chaque coordinateur gère l'activité de son étoile indépendamment du reste de l'activité du réseau. Les échanges durant la période d'activité entre les coordinateurs et les feuilles sont gérés selon le mécanisme CSMA/CA slotté pour le trafic non-contraint, et avec des GTS pour le trafic contraint.

Relais garanti. Un relais de trames de niveau MAC est effectué par MaCARI, pour faire cheminer le trafic de type contraint. Ayant connaissance de l'adresse hiérarchique de la destination finale et des paramètres de la topologie, MaCARI réalise ce relais en suivant l'arbre et en appliquant le protocole de routage hiérarchique durant les intervalles de relais garanti entre père et fils, ceci sans solliciter la couche réseau des entités intermédiaires.

C'est par ce mécanisme que MaCARI remplit son objectif d'offrir deux niveaux de qualité de service : un accès garanti pour le trafic contraint et un accès non-garanti pour le trafic non-contraint. L'accès garanti est réalisé durant les intervalles de relais garanti entre les couples de coordinateurs père-fils, et durant les GTS réservés aux communications entre le coordinateur et ses feuilles. L'accès non-garanti est réalisé durant la période d'activité des étoiles et durant $[T_2; T_3]$ en utilisant l'algorithme de CSMA/CA slotté.

Les deux entités du couple père-fils sont les seules entités du réseau à être actives durant l'intervalle de relais : l'accès au médium est négocié uniquement entre ces deux entités. Pour éviter les collisions entre les trames envoyées par le coordinateur fils et celles envoyées par le coordinateur père, un protocole d'échanges entre ces deux coordinateurs est défini comme suit.

Au début de l'intervalle de relais, le coordinateur fils initie l'échange en envoyant des trames à son père. Les trames sont acquittées une à une. Quand le fils n'a plus de trames à envoyer, il envoie une trame de contrôle indiquant la fin du relais montant. Ceci permet au père de commencer le relais descendant et de faire passer les trames de type contraint à destination de ce fils.

Les intervalles de relais garanti peuvent être considérés comme un mécanisme de TDMA pour lequel chaque coordinateur possède son propre intervalle de temps. En d'autres termes, $[T_1; T_2]$ est divisée en plusieurs intervalles de temps : chaque intervalle de temps est dédié à une étoile pour la collecte de données et le relais des trames.

La figure 3.6 met en évidence le découpage d'un cycle MaCARI pour un arbre binaire de 7 nœuds qui favorise le trafic montant. En effet, pendant $[T_1; T_2]$, les étoiles les plus profondes dans l'arbre ont leur période d'activité en premier. Pendant ces périodes d'activité, le coordinateur de chaque étoile échange des données en CSMA/CA slotté et/ou en GTS avec ses feuilles puis remonte le trafic contraint à son coordinateur père. Ensuite, pendant $[T_2; T_3]$, les coordinateurs échangent le trafic non-contraint. Enfin, tous les nœuds entrent en période d'inactivité jusqu'au début du prochain cycle.

L'acheminement de bout en bout pour le trafic contraint est réalisé *via* les intervalles de relais par l'organisation du séquencement des périodes d'activité des étoiles de l'arbre.

L'exemple vu dans la figure 3.6 présente un séquencement de l'activité des étoiles durant la période $[T_1; T_2]$ basé sur un parcours en largeur inversé de l'arbre. Ce type de configuration est appelé cycle global montant parce qu'il favorise le flux montant du trafic contraint : une trame émise en trafic contraint par une feuille peut être remontée vers le haut de l'arbre (par exemple le CPAN) en un cycle global (sauf dans le cas de saturation du réseau pour ce type de trafic). Cette configuration est utilisée pour les applications de collecte. Inversement, le cycle global descendant repose sur une organisation de l'activité des étoiles basée sur un parcours en largeur de l'arbre : le trafic contraint descendant est favorisé permettant à une trame émise par le CPAN d'atteindre l'étoile de sa destination en un cycle global.

MaCARI peut alterner cycle global montant et cycle global descendant pour rendre les flux de trafic homogènes dans le cas d'applications impliquant des échanges à destination quelconque. L'état montant ou descendant d'un cycle est indiqué dans la balise (codé sur un bit). L'algorithme de calcul du séquencement des périodes d'activité des étoiles pour $[T_1; T_2]$ est simple et facile à réaliser : il repose sur le parcours en largeur ou le parcours en largeur inversé. Cette alternance permet de borner l'acheminement d'une trame non reportée (émise lors du cycle de

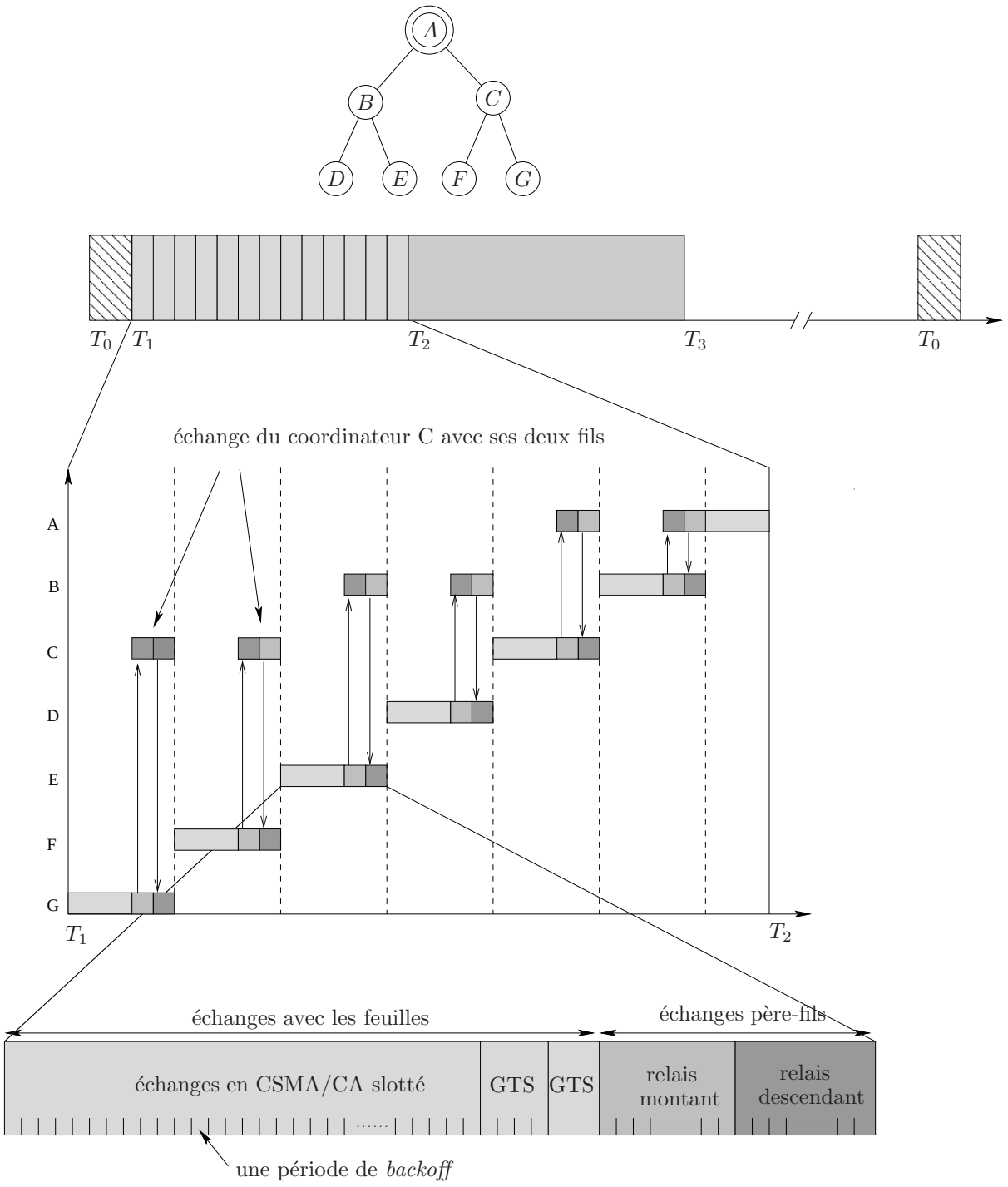


Figure 3.6 – Décomposition d'un cycle MaCARI.

sa génération) de bout en bout. On peut alors borner dans le temps le délai d'acheminement d'une trame selon la durée d'un cycle. Or, le pire des cas correspond à une trame générée par une feuille à la fin de sa période de GTS lorsque la durée restante dans cette période n'est pas suffisante pour envoyer la trame : l'envoi de la trame est donc reporté dans le cycle suivant.

Période de routage. Afin de permettre des communications entre coordinateurs, une période de routage suit la période d'activité des étoiles. Durant cette période de routage, les coordinateurs acheminent les données récoltées durant les différentes périodes d'activités des étoiles (ou communiquent des informations aux actionneurs).

Durant cette période, ce sont uniquement les coordinateurs qui sont actifs et communiquent entre eux en utilisant le CSMA/CA slotté. Dans cette version de la pile protocolaire OCARI, les travaux de l'INRIA sur l'usage de la coloration des nœuds ne sont pas pris en compte [MMG10].

Gestion des files d'attente dans MaCARI

La figure 3.7 représente la gestion des files d'attente d'un coordinateur. Cette représentation montre la classification des paquets dans les files d'attente selon leur type et leur prochain saut. Quand la méthode d'accès MaCARI reçoit un paquet de la couche réseau, elle vérifie le type du paquet. Si le paquet est de type non-contraint et non destiné à une feuille (test numéro 1), il est mis dans la file d'attente appelée routage dans la figure. Cette file d'attente contient tous les paquets à transmettre durant la période $[T_2; T_3]$. Si le paquet est de type contraint ou à destination d'une feuille qui appartient au coordinateur, il est mis dans une file d'attente en fonction du prochain saut (test numéro 2). Si le prochain saut est une feuille du coordinateur, le paquet est mis dans la file d'attente intra-étoile. Sinon, le paquet est mis dans la file d'attente qui correspond au prochain saut, qui est le père ou l'un des n fils.

Ces files d'attente sont utilisées en fonction des périodes de temps de MaCARI. Par exemple, quand le coordinateur est dans l'intervalle de la période d'activité de son étoile, il sollicite la file d'attente intra-étoile. Le trafic à échanger dans chacun des intervalles de temps est classé dans une file d'attente différente. Dans le cas d'une feuille, il n'existe qu'une seule file d'attente qui est la file intra-étoile, puisque les feuilles ne transmettent que durant la période d'activité de leur étoile.

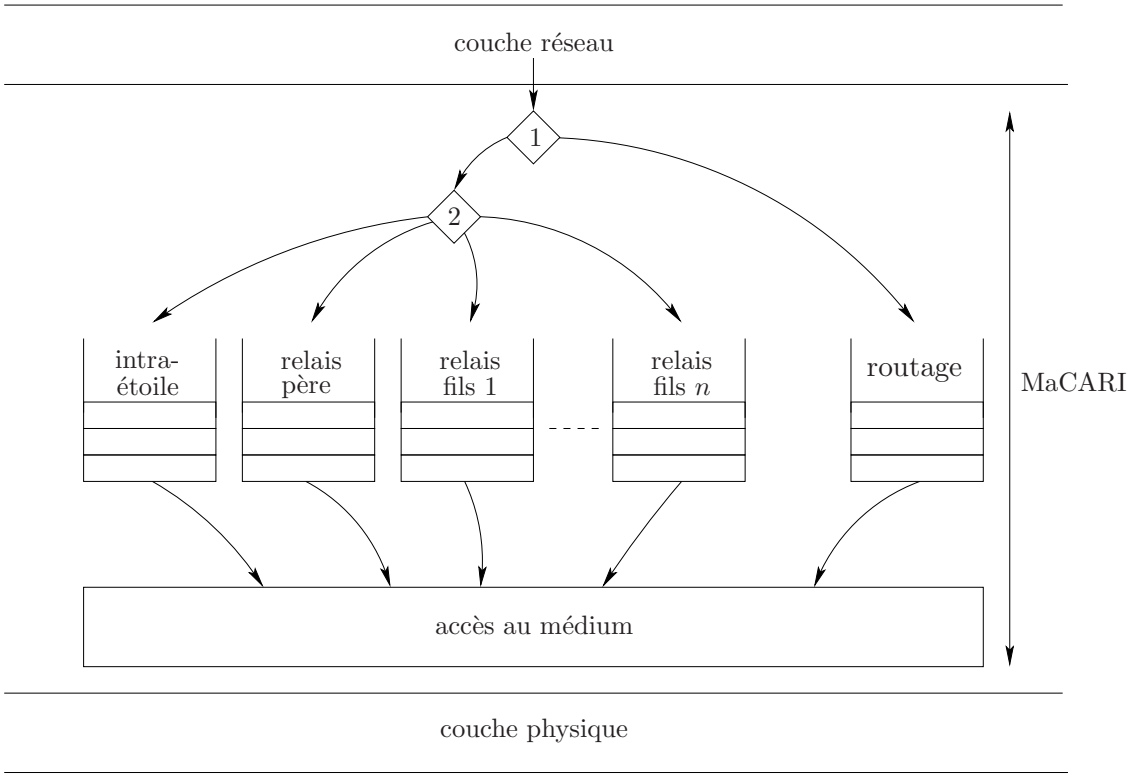


Figure 3.7 – Gestion des files d'attente selon le type de trafic et le prochain saut.

3.2.4 Protocoles de routage de la pile OCARI

Dans cette section, nous détaillons le protocole de routage utilisé dans OCARI, ainsi que son intégration avec la méthode d'accès MaCARI. Ce protocole de routage a été une des contributions du laboratoire INRIA dans le projet OCARI. Ensuite, nous détaillons le protocole de routage qui a été implémenté par le LIMOS pour pouvoir tester MaCARI.

EOLSR/SERENA

EOLSR. EOLSR (pour *Energy efficient extension of OLSR*) [MM08a], est une extension du protocole OLSR utilisant des informations relatives à l'énergie. EOLSR choisit ses MPR, nommés EMPR (pour *Energy efficient MPR*) selon leur énergie résiduelle. Ensuite, les routes sont établies uniquement avec des nœuds EMPR. Le but de EOLSR est de maximiser la durée de vie du réseau en :

- minimisant l'énergie consommée par la transmission d'un paquet de la source jusqu'à la destination,
- répartissant la charge du trafic entre les nœuds du réseau, puisqu'utiliser les mêmes nœuds pour acheminer les paquets épuise les batteries de ces nœuds, et conduit à partitionner le

réseau,

– évitant les nœuds ayant une faible énergie résiduelle,

– réduisant le surcoût en nombre de messages de contrôle.

Le choix des EMPR permet à chaque nœud n du réseau de sélectionner un sous-ensemble de ses voisins à un saut, selon leur énergie résiduelle, qui permet d'atteindre tous les voisins situés à deux sauts en suivant un chemin avec suffisamment d'énergie.

EOLSR est utilisé comme protocole de routage durant $[T_2; T_3]$, la période d'activité non-ordonnancée du cycle global de MaCARI. Durant $[T_1; T_2]$ le protocole de routage hiérarchique est utilisé.

SERENA. SERENA (pour *Scheduling Router Node Activity*) [MM09] permet aux coordinateurs de dormir ou de travailler en parallèle et ainsi d'économiser de l'énergie, tout en garantissant des communications de bout-en-bout. Ceci est réalisé en utilisant le mécanisme de coloration. SERENA est un algorithme décentralisé d'allocation de couleurs aux nœuds pour une réutilisation spatiale du temps. Il est exécuté localement dans chaque nœud. Chaque nœud possède une couleur et chaque couleur est associée à un intervalle de temps distribué par le CPAN *via* MaCARI. Un nœud reste actif durant l'intervalle de temps de sa couleur et durant les intervalles de temps des couleurs de ses voisins à un saut. SERENA est basé sur deux algorithmes :

1. Un algorithme de coloriage à deux sauts [1], qui associe une couleur pour chaque coordinateur du réseau de sorte que deux nœuds qui sont des voisins à un ou deux sauts possèdent des couleurs différentes. Une couleur peut être réutilisée à trois sauts. Le nombre total de couleurs et le nombre de messages échangés pour réaliser la coloration doivent être réduits le plus possible.
2. Un algorithme d'attribution des créneaux, qui attribue un ensemble de créneaux temporels pour chaque coordinateur dans le réseau en se basant sur la couleur. Ces créneaux sont utilisés pour l'envoi des paquets. Chaque nœud doit donc être réveillé dans son créneau, afin de pouvoir transmettre ses propres paquets, et dans les créneaux de ses voisins à un saut, afin de recevoir leurs paquets. Chaque coordinateur peut passer en mode sommeil dans les autres créneaux.

L'attribution des créneaux de temps de SERENA est confiée à MaCARI. SERENA se charge d'acheminer vers le CPAN la liste des couleurs utilisées dans le réseau. Ensuite, le CPAN prend

1. Un algorithme de coloriage à trois sauts si l'acquittement immédiat est utilisé.

en charge le découpage temporel de SERENA et intègre l'intervalle de temps réservé à chaque couleur dans la balise de synchronisation. SERENA est utilisé avec EOLSR durant $[T_2; T_3]$.

Dans cette thèse, nous n'utilisons pas le coloriage des nœuds et le protocole de routage que nous avons utilisé dans la période $[T_2; T_3]$ n'est pas EOLSR. Mais nous avons utilisé le protocole de routage qui a été développé par le LIMOS.

Protocole de routage utilisé dans MaCARI durant $[T_2; T_3]$

Ce protocole de routage a été développé par le LIMOS afin d'évaluer les performances de MaCARI. Durant la période $[T_2; T_3]$, tous les coordinateurs sont actifs et participent au processus de routage des paquets vers la destination finale. Le protocole de routage utilisé dans la période $[T_2; T_3]$ est une amélioration du protocole de routage hiérarchique, qui utilise la table de voisinage à un saut. Cette table est construite durant la période de synchronisation $[T_0; T_1]$, et contient l'adresse courte du coordinateur père, des coordinateurs fils et de chaque coordinateur duquel une balise est reçue. Nous faisons ici l'hypothèse que les liens sont symétriques (éventuellement en considérant uniquement ceux qui sont entendus au delà d'une certaine puissance).

L'algorithme 1 décrit le processus de routage utilisé durant la période d'activité $[T_2; T_3]$. Le prochain saut est calcué en fonction de l'adresse courte de la destination finale du paquet et des voisins. Si l'adresse de la destination finale est l'adresse courte du coordinateur, le paquet est remonté à la couche supérieure. Si cette adresse correspond à l'adresse courte d'une feuille du coordinateur, ou à l'adresse courte d'un coordinateur voisin, l'adresse du prochain saut est l'adresse de la destination finale. Si la destination finale est parmi les descendants du nœud, l'adresse du prochain saut est calculée selon la formule proposée par le protocole hiérarchique. Sinon, si l'un des voisins, qui n'est ni un fils ni le père, a la destination finale dans sa descendance, le prochain saut est l'adresse de ce voisin. Sinon, le paquet est remonté au père.

Ce protocole ne demande aucun échange supplémentaire de messages de contrôle, il ne surcharge donc pas le réseau par un trafic de contrôle. Toutes les informations nécessaires existent déjà dans la couche MaCARI.

Ce protocole de routage est largement inspiré du protocole de routage hiérarchique et du protocole de routage raccourci. Puisque ces deux protocoles de routage sont compatibles, le protocole de routage de MaCARI utilisé durant $[T_2; T_3]$ ne cause jamais de détours dans le réseau quand il est utilisé avec le protocole hiérarchique durant $[T_1; T_2]$. Les échanges entre les différentes files d'attentes de MaCARI ne peuvent donc pas aboutir à des détours dans le réseau. En outre, les paquets attendent moins de temps pour être traités. Par conséquent, les

Algorithme 1 Algorithme de routage appliqué durant la période $[T_2; T_3]$.

```
si l'adresse destination finale est l'adresse courte du nœud courant alors
    remonter le paquet à la couche supérieure
sinon si la destination finale est une feuille du nœud courant alors
    mettre le paquet dans la file d'attente intra-étoile
sinon si la destination finale est un coordinateur voisin du nœud courant alors
    le prochain saut est ce coordinateur
sinon si la destination finale est un descendant du nœud courant alors
    le prochain saut est calculé en utilisant le routage hiérarchique
sinon si la destination finale est un descendant d'un voisin du nœud courant alors
    le prochain saut est ce voisin
sinon
    le prochain saut est le père du nœud courant
fin si
```

performances du réseau augmentent.

Acheminement des données

La figure 3.8 représente l'acheminement des données ainsi que la différenciation de services de la source vers la destination en passant par des nœuds intermédiaires représentés par n_i sur la figure. Dans la période d'activité ordonnancée, le trafic contraint est relayé par la décision de la méthode d'accès MaCARI avec le protocole de routage hiérarchique de ZigBee. Dans la période d'activité non-ordonnancée, le protocole de routage EOLSR est activé. Le trafic non contraint est routé alors par la décision de EOLSR concaténé avec l'algorithme de coloriage SERENA.

3.3 Optimisation des performances pour le trafic contraint

Dans cette partie, nous montrons le problème de dimensionnement du temps pour des réseaux de capteurs sans fil utilisant le mécanisme TDMA. Ensuite, nous proposons une nouvelle approche afin de résoudre ce problème. Nous rappelons que le mécanisme TDMA est convenable pour éviter la collision au niveau du médium et assurer un délai de bout-en-bout faible. Notre approche consiste à utiliser MaCARI avec ses deux mécanismes (TDMA utilisé durant $[T_1; T_2]$ et CSMA/CA partagé par tous les coordinateurs utilisé durant $[T_2; T_3]$) et permettre d'envoyer des trames qui n'ont pas été traitées durant les intervalles TDMA (à cause de la durée limitée de ces intervalles) en utilisant le CSMA/CA. En d'autres termes, notre approche est utilisée pour absorber le trafic accumulé dans les nœuds, $[T_2; T_3]$ servant de soupape au trafic initialement dédié à TDMA.

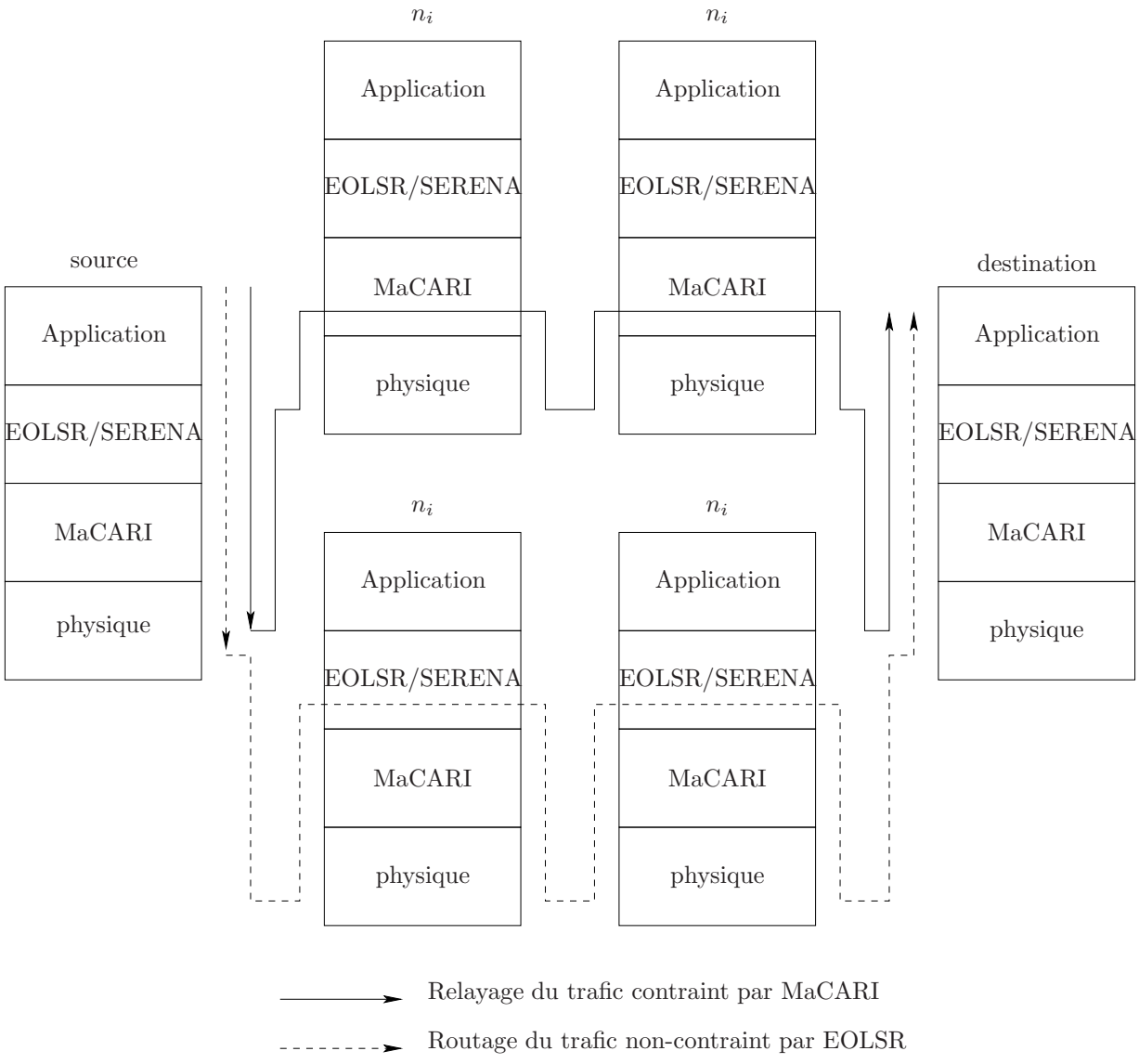


Figure 3.8 – Acheminement des données : différenciation des services.

3.3.1 Description du problème

Dans cette partie, nous montrons le problème d'accumulation du trafic dans le réseau. Nous considérons que la destination est toujours le CPAN. Lors de la transmission de données, le trafic traverse l'arbre pour arriver au CPAN. Par conséquent, la charge du trafic à mesurer augmente à l'approche du CPAN. Un nœud proche du CPAN risque de ne pas avoir le temps de traiter tous les paquets dans sa file d'attente, ce qui va faire que celle-ci risque d'être remplie et que ce nœud perde des paquets.

La figure 3.9 illustre un exemple d'accumulation du trafic dans le réseau MaCARI. Comme nous l'avons déjà expliqué, dans MaCARI, les coordinateurs récoltent les données de leurs feuilles durant la période intra-étoile et les acheminent durant la période de relais garanti. Les flèches sur la figure représentent le flux de données traversant le réseau durant la période $[T_1; T_2]$. Nous remarquons qu'en s'approchant du CPAN, qui est la destination des flux de données, les flèches deviennent plus nombreuses. En effet, un coordinateur à une profondeur d concatène les trames de ses feuilles ainsi que les trames de ses coordinateurs fils de profondeur supérieure ou égale à $d + 1$, afin de les transmettre à son père de profondeur $d - 1$.

3.3.2 Description de la solution

Dans cette partie, nous proposons un mécanisme hybride pour soulager les intervalles TDMA. Ce mécanisme consiste, à non seulement utiliser MaCARI, mais aussi permettre aux trames qui surchargent la période TDMA d'être transmises durant la période CSMA/CA. Cependant, la compétition pour l'accès au médium durant la période CSMA/CA est forte, puisque tous les nœuds du réseau sont actifs durant cette période.

La figure 3.10 montre la différence entre l'utilisation du mécanisme TDMA classique (qui est référencée comme l'approche TDMA seul), comme illustré dans la partie (a), et l'utilisation du mécanisme hybride de MaCARI, comme illustré dans la partie (b). Le trafic est stockée sous forme de trames dans la file d'attente avant d'être traité. Quand l'approche TDMA seul est utilisée, les trames qui ne peuvent pas être transmises durant leur intervalle de temps doivent attendre l'occurrence de leur prochain intervalle de temps, qui aura lieu durant le prochain cycle global. Donc, les trames qui n'ont pas pu être transmises sont retardées. Quand l'approche hybride de MACARI est utilisée, les trames peuvent être transmises durant leur intervalle de temps ou durant la période CSMA/CA.

Dans l'exemple de la figure 3.10(b), nous considérons que trois trames ont le temps d'être transmises durant la période TDMA. Durant le second cycle global, cinq trames sont stockées

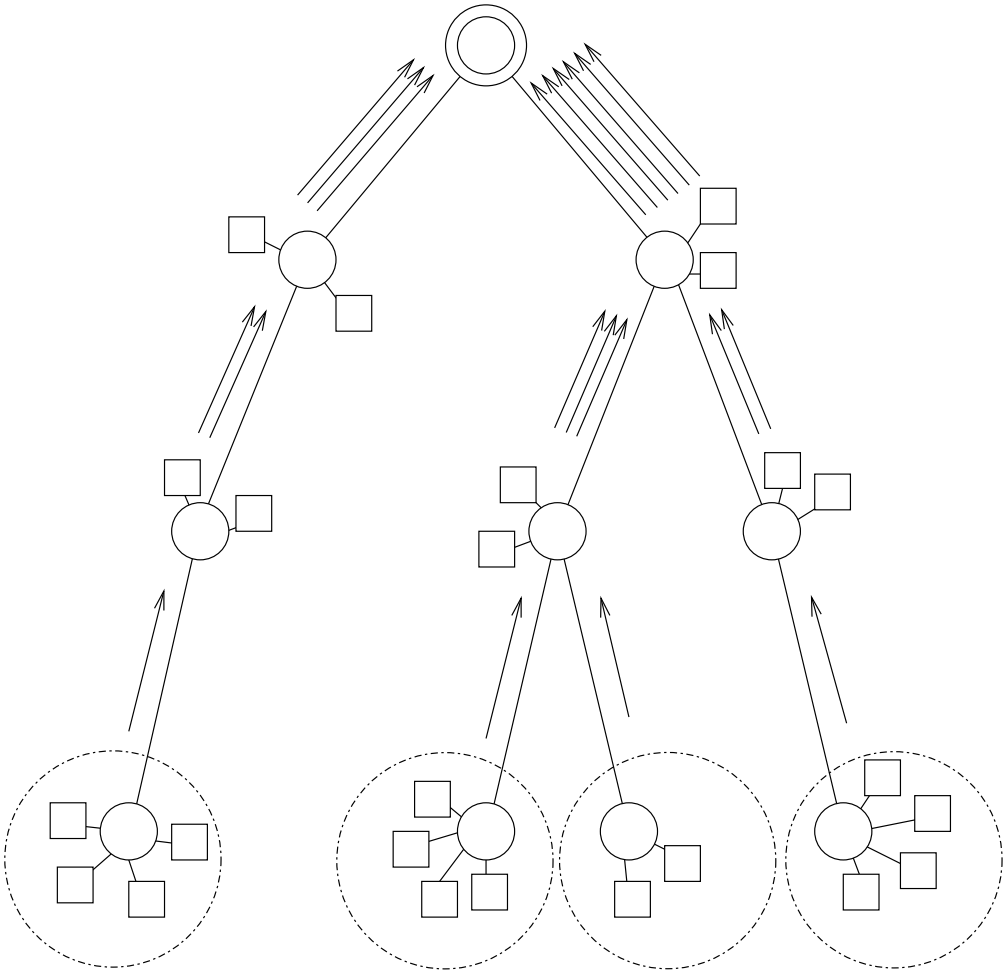


Figure 3.9 – Exemple d'accumulation du trafic contraint.

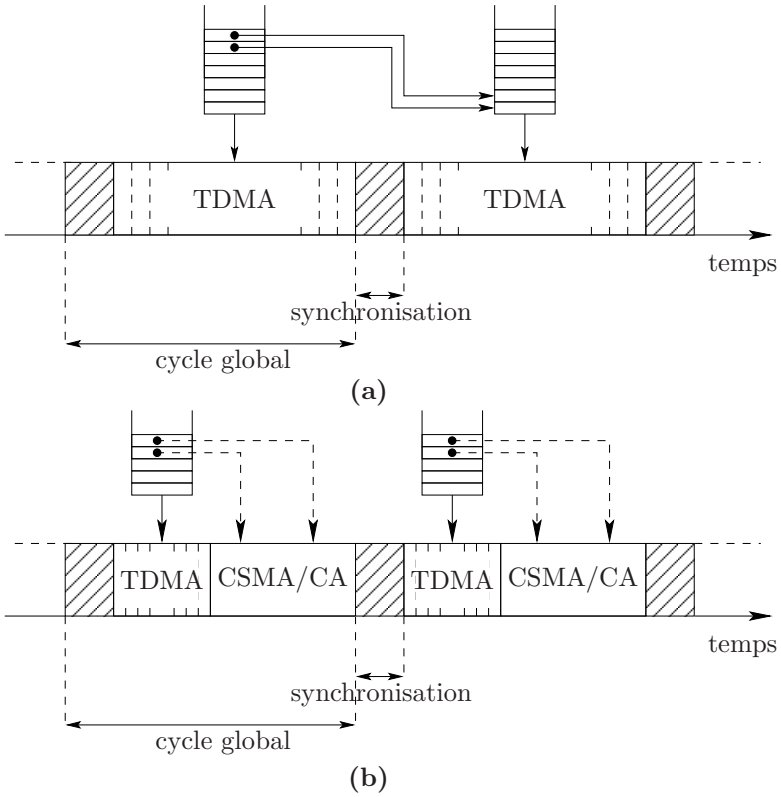


Figure 3.10 – Ajout d'une période CSMA/CA après une période TDMA.

dans la file d'attente : trois sont transmises durant les intervalles TDMA, et les deux trames qui restent sont transmises en utilisant le protocole CSMA/CA. Dans la figure 3.10(a), plus de trames peuvent être traitées puisque les intervalles TDMA sont plus grands.

Il est important de mentionner que, si une trame est envoyée durant la période CSMA/CA et n'est pas reçue par son prochain saut (c'est-à-dire que l'émetteur n'a pas reçu d'acquittement), elle est réinjectée à sa position initiale dans la file d'attente. Ainsi, la trame aura l'opportunité d'être correctement envoyée et reçue dans son prochain intervalle TDMA. En effet, les trames du trafic contraint ne doivent pas être perdues à cause de la compétition, dans la période CSMA/CA.

Avec ce mécanisme, les trames attendent moins de temps pour être traitées que si l'on utilise le mécanisme TDMA seul. Par conséquent, avoir un grand trafic rend le délai plus important dans l'approche TDMA seul que dans notre approche hybride.

3.4 Optimisation des performances pour le trafic contraint et le trafic non-contraint

Dans cette partie, nous avons décidé d'appliquer le mécanisme des échanges de files d'attente sur l'architecture du réseau OCARI pour montrer les bénéfices de ce mécanismes dans un réseau industriel. Avec ce mécanisme, nous visons une amélioration des performances en termes de délai, de taux de pertes et de débit sans augmenter la consommation de l'énergie dans le réseau. Le mécanisme des échanges est appliqué seulement sur les files d'attente $\mathcal{Q}_i$ des coordinateurs du réseau. Les feuilles ne sont pas concernées.

3.4.1 Échanges de files d'attente dans OCARI

Dans cette partie, nous détaillons le mécanisme des échanges de paquets entre les files d'attente dans OCARI.

L'architecture du réseau OCARI est une architecture multi-couches. Les protocoles de routage utilisés sont EOLSR/SERENA et le protocole de routage hiérarchique. Les méthodes d'accès utilisées sont TDMA et CSMA/CA. Ces protocoles sont classés en deux combinaisons et à un instant donné une seule combinaison est active. En effet, durant la période $[T_1; T_2]$, la combinaison TDMA-hiérarchique est active. Cette combinaison assure la bonne réception du trafic contraint. Durant la période $[T_2; T_3]$, la combinaison CSMA/CA-EOLSR est active. Cette période est utilisée pour router le trafic non-contraint.

Cette architecture est montrée sur la figure 3.11, quand le mécanisme des échanges de files d'attente est activé. Les flèches pleines correspondent à l'acheminement de base dans chaque pile protocolaire, alors que les flèches en pointillées correspondent à l'acheminement lors des échanges des files d'attente. En effet, le mécanisme autorise l'envoi du trafic non-contraint durant $[T_1; T_2]$ mais en respectant les conditions de cette période. En d'autres termes, le routage hiérarchique recalcule le prochain saut selon sa fonction et la méthode d'accès au médium est le TDMA. Une trame du trafic non-contraint ne peut être envoyée durant $[T_1; T_2]$ que lorsque la file d'attente de cette période est vide. De même, le mécanisme autorise l'envoi du trafic contraint durant $[T_2; T_3]$ en respectant les conditions de cette période. Ainsi le protocole de routage utilisé est EOLSR et la méthode d'accès au médium est le CSMA/CA.

La figure 3.12 illustre une autre représentation du mécanisme d'échanges de paquets des files d'attente. Ce mécanisme permet de récupérer le temps de la période associée à la combinaison $(\mathcal{M}_i, \mathcal{R}_i)$ activée et l'utiliser pour envoyer du trafic dédié à une autre combinaisons $(\mathcal{M}_j, \mathcal{R}_j)$

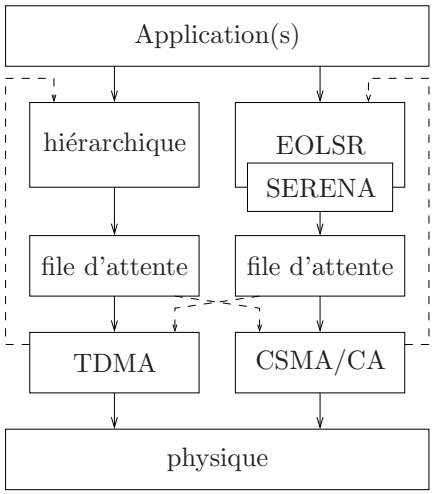


Figure 3.11 – Architecture de la pile d'OCARI avec échanges de files d'attente.

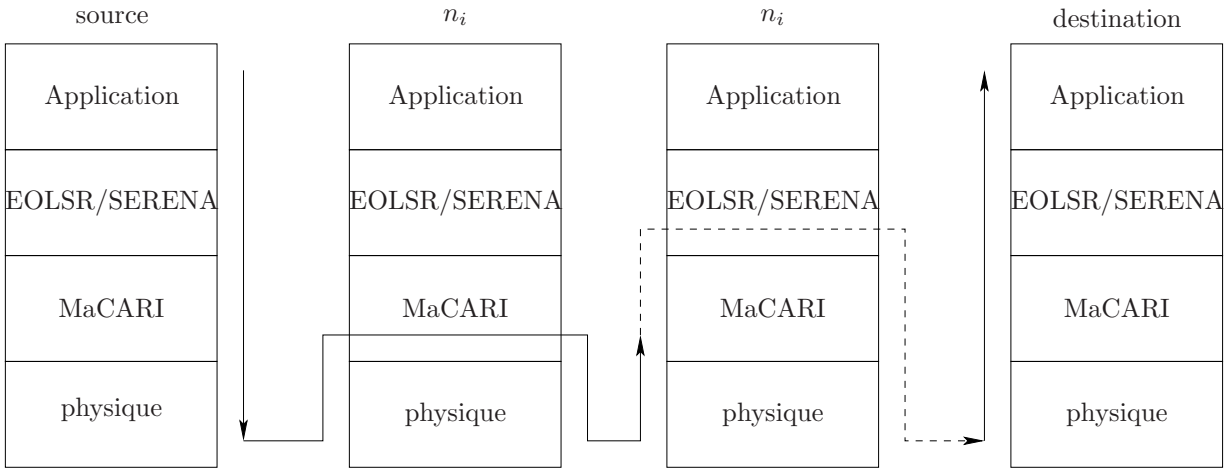
pour tout $j \neq i$. Les échanges peuvent avoir lieu dans n'importe quel nœud tant que la file d'attente associée à la combinaison active est vide. La figure 3.12(a) montre les échanges du trafic contraint représenté par des flèches pleines. La figure 3.12(b) montre les échanges du trafic non-contraint représenté par des flèches en pointillées.

3.4.2 Apparition de détours lors des échanges de files d'attente

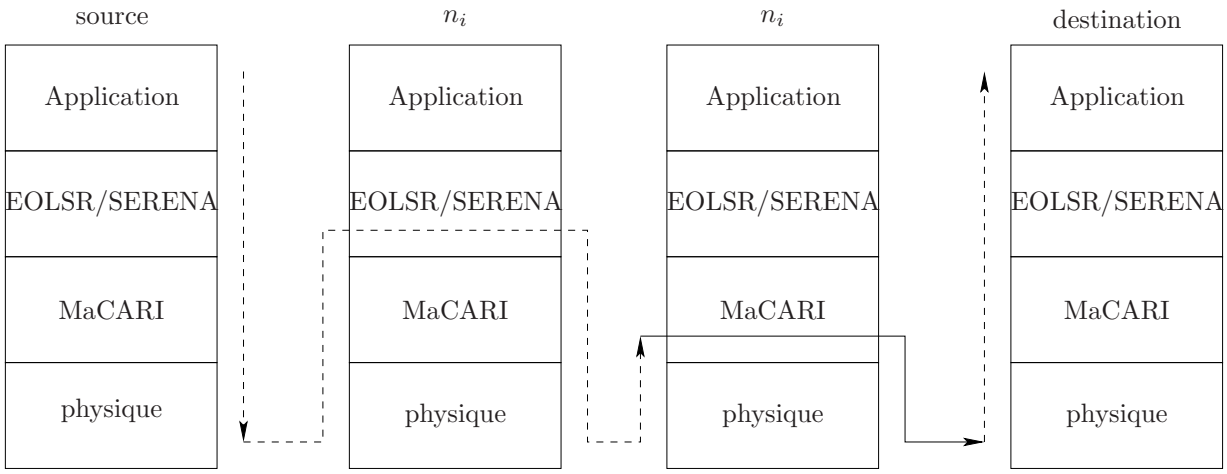
Le fait d'échanger des paquets entre les différentes files d'attente risque de faire apparaître des détours dans le réseau durant l'acheminement des paquets. Dans cette partie nous détaillons ce risque et nous montrons comment l'éviter.

Les échanges de files d'attente dans OCARI peuvent aboutir à des détours dans le réseau. En effet, dans la période $[T_1; T_2]$, seules les communications père-fils sont considérées et ceci en utilisant le protocole hiérarchique. Dans la période $[T_2; T_3]$, le protocole de routage EOLSR est utilisé. Quand le protocole hiérarchique cohabite avec le protocole OLSR, il peut générer des détours dans le réseau (voir théorème 3 du chapitre 2). Une solution est de rendre ces protocoles retardables en utilisant une fonction de conservation.

Les échanges de files d'attente dans MaCARI se font entre deux combinaisons $(\mathcal{M}_i, \mathcal{R}_i)$ différentes de celles de la pile protocolaire d'OCARI. En effet, durant la période d'activité $[T_1; T_2]$, nous concentrons notre étude sur les intervalles de relais montant et descendant, puisque la période intra-étoile ne permet pas de faire du routage. Le protocole de routage utilisé dans cette période est le routage hiérarchique : seules les communications père-fils sont autorisées dans ces intervalles. Dans la période d'activité $[T_2; T_3]$, le protocole de routage utilisé est une version optimisée du protocole de routage hiérarchique. Comme nous l'avons déjà mentionné, ce



(a) Échange du trafic contraint



(b) Échange du trafic non-contraint

Figure 3.12 – Acheminement du trafic contraint et du trafic non-contraint en appliquant le mécanisme d'échange de files d'attente dans OCARI.

protocole de routage utilisant une table de voisinage permet de raccourcir l'arbre selon l'algorithme 1 expliqué dans ce chapitre. Ce protocole de routage est largement inspiré du protocole de routage hiérarchique et du protocole de routage raccourci. Puisque ces deux protocoles de routage (hiérarchique et raccourci) sont compatibles, le protocole de routage de MaCARI utilisé durant $[T_2; T_3]$ ne cause jamais de détours dans le réseau quand il est utilisé avec le protocole hiérarchique durant $[T_1; T_2]$. La compatibilité de ces deux protocoles est directement prouvée en se basant sur le théorème 1 du chapitre 2. Les échanges entre les différentes files d'attentes de MaCARI ne peuvent donc pas aboutir à des détours dans le réseau. En outre, les paquets attendent moins de temps pour être traités. Par conséquent, les performances du réseau augmentent.

3.4.3 Échanges de files d'attente dans MaCARI

Dans cette partie, nous expliquons le mécanisme d'échanges entre les différentes files d'attente dans MaCARI. Ce mécanisme a pour but d'optimiser les performances du réseau.

Mécanisme

Le mécanisme des échanges consiste à envoyer des paquets du trafic contraint (respectivement trafic non-contraint) dans la période interprétant le trafic non-contraint (respectivement le trafic contraint). Ceci se fait seulement dans le cas où il n'y a plus de paquets du trafic initialement prévu pour la période considérée. Ce mécanisme permet de réduire le délai de bout-en-bout et augmenter le débit sans augmenter la consommation d'énergie des nœuds.

Scénario d'échanges des files d'attente dans MaCARI

Dans le mécanisme de base de MaCARI, la période $[T_1; T_2]$ gère le trafic contraint. Dans cette période, un nœud possède une file d'attente pour son père, une file d'attente pour ses feuilles, et une file d'attente pour chacun de ses coordinateurs fils. La période $[T_2; T_3]$ gère le trafic non-contraint. Dans cette période, une seule file d'attente est nécessaire.

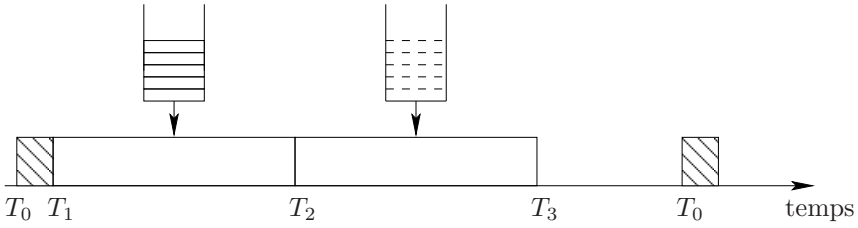
Notre mécanisme intervient à la fin de chaque phase de traitement usuel des trames. Chaque période applique ses propres caractéristiques (calcul du prochain saut au niveau routage et activation des nœuds au niveau MAC) pour l'acheminement de la trame. En effet, pour un instant $t \in [T_1; T_2]$, si la file d'attente dédiée à cet instant de cette période est vide (la file d'attente du relais père ou la file d'attente d'un des relais fils), notre algorithme examine la file d'attente de la période $[T_2; T_3]$ et en extrait certains de ses paquets un à un (le nombre

de paquets traités dépend de la durée restante dans la période $[T_1; T_2]$) pour les envoyer en respectant les critères de la période $[T_1; T_2]$. Notre mécanisme nécessite le recalcul du prochain saut en se basant sur le protocole de routage actif dans $[T_1; T_2]$. Le recalcul du prochain saut est important pour le réseau. En effet, quand un paquet change de période, la méthode d'accès change et l'état de chaque nœud voisin risque de changer, et par suite, le nœud qui a été sélectionné comme prochain saut pour le paquet en cours de transmission durant la période initiale peut être en mode sommeil durant la période courante ; le paquet doit donc attendre le réveil du nœud pour qu'il puisse être envoyé. Le mécanisme des échanges ne réduit pas le délai dans ce cas. Si le paquet n'a pas pu être envoyé, notre mécanisme remet le paquet dans son état initial à la place qu'il occupait dans sa file d'attente d'origine.

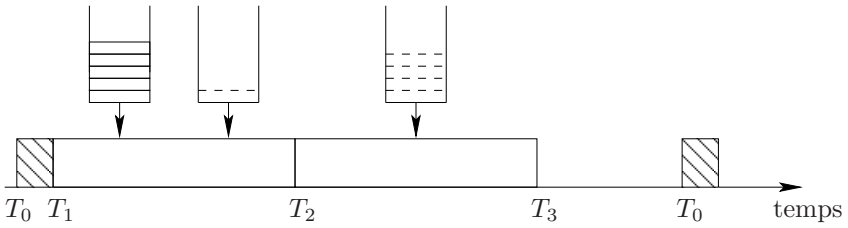
L'algorithme 2 illustre le mécanisme des échanges des paquets de la période $[T_2; T_3]$ à la période $[T_1; T_2]$. L'échange des paquets entre les files d'attente se fait à chaque fois que l'ensemble des files d'attente de la période $[T_1; T_2]$ est vide. Notre mécanisme ne considère pas les paquets de la file d'attente intra-étoile puisque ces paquets sont générés par les feuilles, qui dorment dans la période $[T_2; T_3]$.

Les échanges de la période $[T_1; T_2]$ à la période $[T_2; T_3]$ sont plus compliqués que ceux de la période $[T_1; T_2]$ puisque notre mécanisme peut extraire les paquets de plusieurs files d'attente. Quand la file d'attente de la période $[T_2; T_3]$ est vide, la file d'attente du père est examinée. À chaque fois qu'un paquet est extrait de cette file d'attente, le prochain saut est recalculé selon le protocole de routage de la période en cours. Si notre algorithme arrive à envoyer tous les paquets de la file d'attente du père, notre mécanisme examine la file d'attente du coordinateur fils ayant le plus grand nombre de paquets à envoyer. Ce processus est répété jusqu'à ce que tous les paquets soient envoyés, ou que le réseau ne soit plus dans $[T_2; T_3]$. L'algorithme 3 illustre le phénomène des échanges dans la période $[T_2; T_3]$.

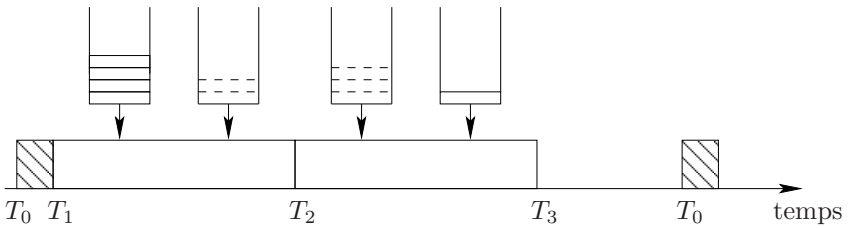
La figure 3.13 montre notre mécanisme d'échanges de files d'attente dans MaCARI selon trois scénarios : les échanges de files d'attente ne sont pas activés, les échanges de files d'attente sont activés dans un seul sens, et les échanges de files d'attente sont activés dans les deux sens. La figure 3.13(a) montre le comportement normal de MaCARI quand les échanges ne sont pas autorisés. Les trames sont traitées durant leurs périodes correspondantes. Pour des raisons de simplicité, nous considérons que chaque période est capable d'envoyer en moyenne 6 trames (bien que ce nombre puisse varier selon la période de *backoff* choisie par le CSMA/CA et la charge locale du réseau). Les trames du trafic contraint (représentées par des lignes pleines) sont envoyées durant $[T_1; T_2]$ seulement : si elles sont produites après l'instant T_2, elles doivent



(a) Pas d'échanges.



(b) Échange dans un sens : $[T_2; T_3] \Rightarrow [T_1; T_2]$.



(c) Échanges dans les deux sens.

Figure 3.13 – Échanges de files d'attente dans MaCARI.

attendre pour le prochain instant T_1 pour être envoyée, ce qui augmente le délai. Les trames du trafic non-contraint (représentées par des lignes en pointillées) subissent un retard similaire si elles sont produites après T_3. La figure 3.13(b) montre le comportement de MaCARI quand les échanges de files d'attente sont autorisés dans un seul sens, plus précisément quand les trames du trafic non-contraint peuvent être aussi envoyées dans $[T_1; T_2]$. On remarque que dans ce cas, le délai du trafic non-contraint est réduit, ainsi que la congestion du médium durant $[T_2; T_3]$ parce que le nombre de trames envoyées diminue. La figure 3.13(c) montre le mécanisme des échanges de files d'attente dans les deux sens. Dans ce cas, les trames peuvent être envoyées durant n'importe quelle période. On remarque que ce scénario réduit le délai de bout-en-bout pour les deux trafics et augmente le débit dans le réseau.

3.5 Conclusion

Les architectures multi-couches permettent d'intégrer une différenciation de QoS dans un réseau de capteurs sans fil industriel. Dans ce chapitre, nous avons proposé d'implémenter un mécanisme d'échanges de paquets entre les files d'attente d'un même nœud dans le projet OCARI. Ce mécanisme est utilisé afin d'améliorer les performances de ce réseau industriel en réduisant le délai et en augmentant le débit.

Nous avons tout d'abord présenté le projet OCARI. Nous avons mentionné les différents partenaires ayant travaillé sur ce sujet. Nous nous sommes concentrés sur MaCARI, la sous-couche MAC d'OCARI. Ensuite, nous avons montré que les protocoles de routage utilisés dans l'implémentation actuelle de MaCARI étaient compatibles et ne peuvent pas aboutir à des détours.

La stratégie que nous avons considérée consiste à permettre l'activation des échanges dans un seul sens ou dans les deux sens entre les trames en instance d'émission pendant $[T_1; T_2]$ et $[T_2; T_3]$. Notre objectif est de montrer des meilleures performances en termes de débit, de taux de pertes et de délai par rapport à une stratégie sans échange.

Algorithme 2 Mécanisme d'échanges des paquets de la file d'attente de la période $[T_2; T_3]$ vers la file d'attente de la période $[T_1; T_2]$, pour un coordinateur donné ayant un paquet à envoyer.

Paramètres : t instant courant, n_c nombre de coordinateurs fils, $\mathcal{Q}(i)$ file d'attente du i-ème coordinateur fils, $\mathcal{Q}(p)$ file d'attente du relais père, $\mathcal{Q}(routage)$ file d'attente de la période $[T_2; T_3]$.

```
1: si t ∈ [T1;T2] alors
2:   si t ∈ période intra-étoile alors
3:     si relais montant actif alors
4:       si il existe un coordinateur fils i tel que Q(i) ≠ ∅ alors
5:         extraire paquet de Q(i)
6:         envoyer paquet
7:       sinon
8:         (* faire l'échange *)
9:         extraire paquet de Q(routage)
10:        calculer prochain saut selon [T1;T2]
11:        si prochain saut actif alors
12:          envoyer paquet
13:        fin si
14:      fin si
15:    sinon
16:      si relais descendant actif alors
17:        si Q(p) ≠ ∅ alors
18:          extraire paquet de Q(p)
19:          envoyer paquet
20:        sinon
21:          si Q(routage) ≠ ∅ alors
22:            (* faire l'échange *)
23:            extraire paquet de Q(routage)
24:            calculer prochain saut selon [T1;T2]
25:            si prochain saut actif alors
26:              envoyer paquet
27:            fin si
28:          fin si
29:        fin si
30:      fin si
31:    fin si
32:  fin si
33: fin si
```

Algorithme 3 Mécanisme d'échanges des paquets de la file d'attente de la période $[T_1; T_2]$ vers la file d'attente de la période $[T_2; T_3]$, pour un coordinateur donné ayant un paquet à envoyer.

Paramètres : t instant courant, n_c nombre de coordinateurs fils, $\mathcal{Q}(i)$ file d'attente du i-ème coordinateur fils, $\mathcal{Q}(p)$ file d'attente du relais descendant, $\mathcal{Q}(routage)$ file d'attente de la période $[T_2; T_3]$.

```
1: si t ∈ [T2;T3] alors
2:   si Q(routage) ≠ ∅ alors
3:     extraire paquet de Q(routage)
4:     envoyer paquet
5:   sinon
6:     (* faire l'échange *)
7:     si Q(p) ≠ ∅ alors
8:       extraire paquet de Q(p)
9:       calculer prochain saut selon [T2;T3]
10:      si prochain saut actif alors
11:        envoyer paquet
12:      fin si
13:    sinon
14:      f ← le fils i ayant la plus grande file d'attente
15:      taille ← taille de Q(f)
16:      si taille > 0 alors
17:        extraire paquet de Q(f)
18:        calculer prochain saut selon [T2;T3]
19:        si prochain saut actif alors
20:          envoyer paquet
21:        fin si
22:      fin si
23:    fin si
24:  fin si
25: fin si
```

Chapitre 4

Résultats

Dans ce chapitre, nous regroupons les résultats de simulations des contributions détaillées dans les chapitres 2 et 3. Nous montrons, tout d'abord, les bénéfices d'un réseau de capteurs sans fil exploitant une architecture multi-couches. Puis, nous étudions le mécanisme d'échange des files d'attente qui permet d'améliorer les performances de l'architecture multi-couches. Pour ce faire, nous procédons en deux temps.

- Nous étudions le trajet d'un paquet dans un réseau peu chargé en faisant des hypothèses simplificatrices [1] pour faciliter l'étude. Ces hypothèses peuvent être vues comme un cas limite et idéal. Cette étude identifie le risque d'apparition de détours pour certains protocoles de routage. Toutefois, nous montrons qu'avec la méthode d'accès MaCARI, les échanges peuvent être réalisés sans générer de détours dans le réseau.
- Nous étudions les échanges des files d'attente dans MaCARI. Les hypothèses de cette étude sont plus réalistes puisqu'elles incluent des files d'attente de taille limitée, une méthode d'accès complète, un médium respectant le modèle ITU, et un modèle de trafic pour la charge soumise au réseau. Nous évaluons les performances de MaCARI en testant successivement un seul type de trafic, puis deux types de trafic.

1. Ces hypothèses concernent la charge du trafic : il n'y a pas de trame en instance dans les files d'attente, la méthode d'accès : elle ne génère pas de collisions et chaque trame est transmise au voisin en un temps constant (la longueur de la trame n'est pas prise en compte), et le médium : il ne produit aucune interférence et aucune perte.

4.1 Évaluation des performances de l'architecture multi-couches

Dans cette partie, nous évaluons par simulation les bénéfices de notre architecture multi-couches. Nous comparons deux combinaisons de protocoles MAC-routage afin de pouvoir accorder les meilleures conditions de communications dans le réseau pour tout type de trafic et par suite réduire la congestion dans le réseau. Tout d'abord, nous présentons les paramètres de simulations utilisés pour extraire les résultats. Ensuite, nous étudions le comportement de plusieurs métriques (comme le taux de pertes, le débit, et le délai) pour cette architecture.

4.1.1 Paramètres de simulations

Les simulations sont réalisées en utilisant le simulateur réseau NS2 [NS202], version 2.31 (largement enrichi par des contributions de notre équipe). Les paramètres de la couche physique sont les suivants : la puissance de transmission est fixée à -25 dBm, le seuil de réception est fixé à -92 dBm, et le seuil de détection de porteuse est choisi égal à -95 dBm. Le modèle de propagation utilisé est le modèle *two ray ground* (avec ses paramètres par défaut). Ces paramètres correspondent à une portée d'environ 25 m.

Nous considérons un réseau en grille composé de 49 nœuds uniformément distribués et espacés de 10 m. Tous les nœuds sont des FFD. La figure 4.1 représente la topologie du réseau testé. Le coordinateur de PAN est situé au centre du réseau. Les paramètres du réseau sont définis comme suit : $R_m = 5$, $C_m = 5$ et $L_m = 5$. La distance maximale entre un FFD et son père est de 20 m, pour assurer que les associations soient réalisées en utilisant des liens de haute qualité. Les liens de la figure correspondent aux liens de l'arbre.

Chaque source génère 30 octets de trafic périodique, une fois que toutes les associations sont réalisées. Chaque simulation est lancée pour 100 secondes, et les nœuds consomment 15 secondes afin de s'associer au réseau. La surcharge due aux entêtes est de 10 octets par trame. Les nœuds sources et destinations sont choisis aléatoirement pour chacune des 100 répétitions. Dans toutes les simulations, nous faisons varier le taux de transmission de données de 1 paquet par seconde et par source jusqu'à 15 paquets par seconde et par source. La taille des files d'attente des nœuds est fixée à 50 paquets.

Nous étudions trois métriques : le taux de pertes, le débit et le délai de bout-en-bout. Le taux de pertes est défini par le rapport du nombre de paquets reçus par la destination finale sur le nombre de paquets générés par le nœud source. Le taux de pertes prend donc en considération les pertes dues aux collisions et aux débordements de files d'attente. Le débit est

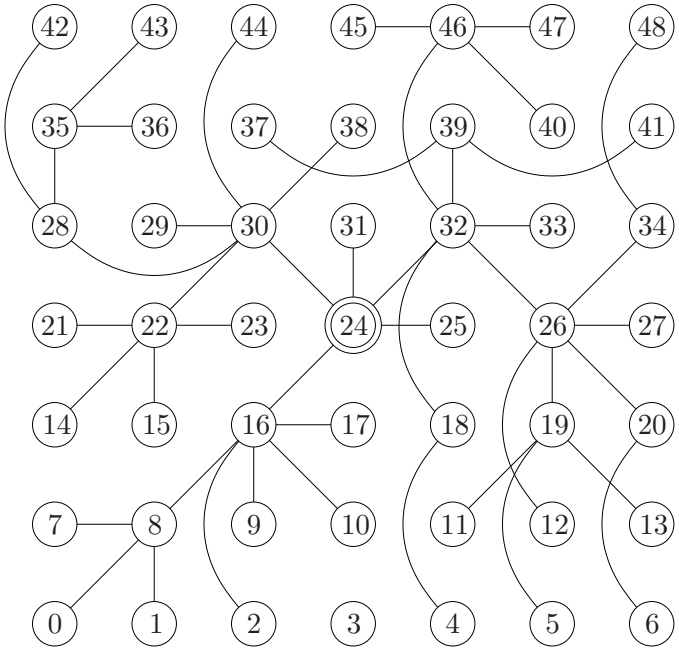


Figure 4.1 – Topologie de 49 nœuds.

défini par la quantité de données utiles reçue par la destination. Le délai de bout-en-bout est défini par l'intervalle de temps moyen entre la première transmission d'un paquet par la source et la réception de ce paquet par la destination. Le délai de bout-en-bout prend en compte les paquets reçus uniquement.

Dans nos simulations, tous les nœuds du réseau sont supposés être parfaitement synchronisés et ont une horloge parfaite (c'est-à-dire sans dérive). Le temps est divisé en un cycle composé de deux intervalles de temps de $4,9$ secondes chacun. Dans chacun de ces intervalles de temps, une combinaison de protocoles MAC-routage est activée. Le premier intervalle de temps utilise la méthode d'accès CSMA/CA slotté et le protocole de routage hiérarchique de ZigBee (pour la topologie arbre) h, et le second intervalle de temps utilise une version de la méthode d'accès TDMA (avec un ordonnancement montant) et le protocole de routage raccourci, r.

Chaque combinaison est activée pour 50% du temps de la simulation. Deux sources produisent un trafic pour la première combinaison (CSMA/CA slotté et routage hiérarchique), et deux autres applications sources produisent du trafic pour la seconde combinaison (TDMA avec ordonnancement montant et routage raccourci).

4.1.2 Taux de pertes

La figure 4.2 montre le comportement du taux de pertes de paquets en fonction du taux de génération des paquets. Pour la première combinaison, et plus généralement pour les protocoles

MAC qui sont des variantes de CSMA/CA, une accumulation du trafic aboutit à des surcharges au début de l'activité [HGM09]. En effet, le trafic généré pendant la période d'inactivité ne peut pas être traité immédiatement par la méthode d'accès et s'accumule dans les files d'attente. Ce trafic est traité au début de la prochaine période d'activité ce qui aboutit à une forte contention dans le médium. Ceci augmente le taux de pertes des paquets. Pour la deuxième combinaison, et plus généralement pour les protocoles MAC basés sur du TDMA, les files d'attente sont rapidement remplies par le trafic cumulé (puisque chaque intervalle de temps est court vis-à-vis du délai offert), ce qui augmente le taux de pertes des paquets. Notons que les files d'attente sont de taille 50 paquets et que, par exemple, à un taux de génération de 2 paquets par seconde et par source, correspond à un nombre de paquets générés de 40 paquets par cycle. La première combinaison est moins performante à faible charge que la seconde puisqu'elle tente de transmettre plus de trames que la deuxième combinaison et donc cette combinaison génère plus de collision que la deuxième combinaison. Pour de faibles taux de transmission, la deuxième combinaison est préférable à la première. Donc, pour des réseaux à faible charge, les simulations montrent qu'il est préférable d'utiliser du TDMA au niveau MAC et du routage raccourci au niveau routage. Pour des réseaux à forte charge, combiner le mécanisme CSMA/CA slotté avec le routage hiérarchique est plus préférable si nous considérons uniquement le taux de perte comme critère.

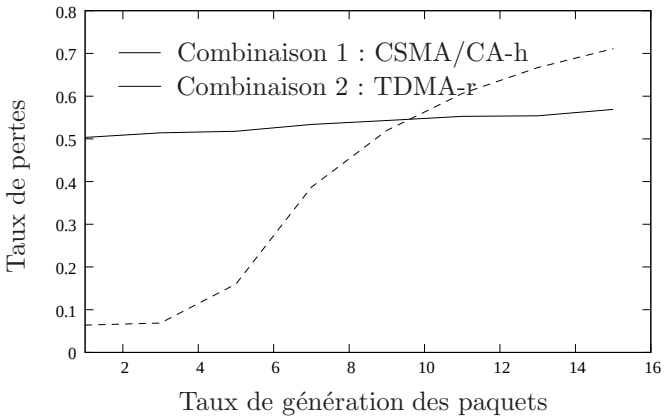


Figure 4.2 – Taux de pertes en fonction de taux de génération des paquets.

4.1.3 Débit

La figure 4.3 montre le débit moyen en fonction du taux de génération des paquets. Pour la première combinaison, le débit augmente avec la charge offerte par les applications, et est loin d'atteindre le seuil de saturation pour une génération de 16 paquets par source et par

seconde. Pour la deuxième combinaison, la saturation est atteinte rapidement, après 5 paquets par seconde et par source. Agrandir la taille des files d'attente des nœuds augmente le débit de la deuxième combinaison, jusqu'à une certaine limite.

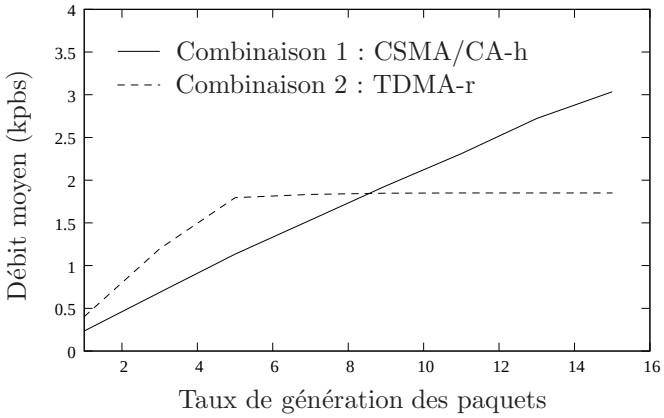


Figure 4.3 – Débit moyen en fonction du taux de génération des paquets.

4.1.4 Délai de bout-en-bout

La figure 4.4 illustre le délai de bout-en-bout moyen en fonction du taux de génération des paquets. Pour la première combinaison, nous remarquons que le délai est presque stable. Notons que la première combinaison montre un grand taux de pertes (environ 50%, cf. figure 4.2) à cause de la surcharge des files d'attente et de la congestion qui a eu lieu au début de l'intervalle de temps de cette combinaison. Ainsi, les paquets perdus sont essentiellement ceux qui ont été accumulés, et qui avaient un grand délai. Les paquets reçus sont principalement ceux qui sont générés pendant l'intervalle de temps de la première combinaison, et le délai moyen qui en résulte est faible. La deuxième combinaison montre un grand délai qui augmente avec le taux de génération du trafic. Ceci est dû à la longueur du cycle qui est $2 \times 4,9 = 9.8$ secondes et au fait que chaque nœud possède un intervalle de temps TDMA de $0,1$ s.

4.1.5 Bilan

La figure 4.2 et la figure 4.3 montrent l'utilité de notre architecture multi-couches qui permet de faire cohabiter plusieurs combinaisons de protocoles MAC-routage. En effet, dans nos simulations, cette architecture apporte un gain en performances globales du réseau pour tout type de trafic (compromis entre taux de pertes et débit) plus important que celui de l'architecture mono-couche qui privilégie un type de trafic au détriment d'un autre.

Le principal inconvénient de l'architecture multi-couches est qu'elle aboutit à de grands

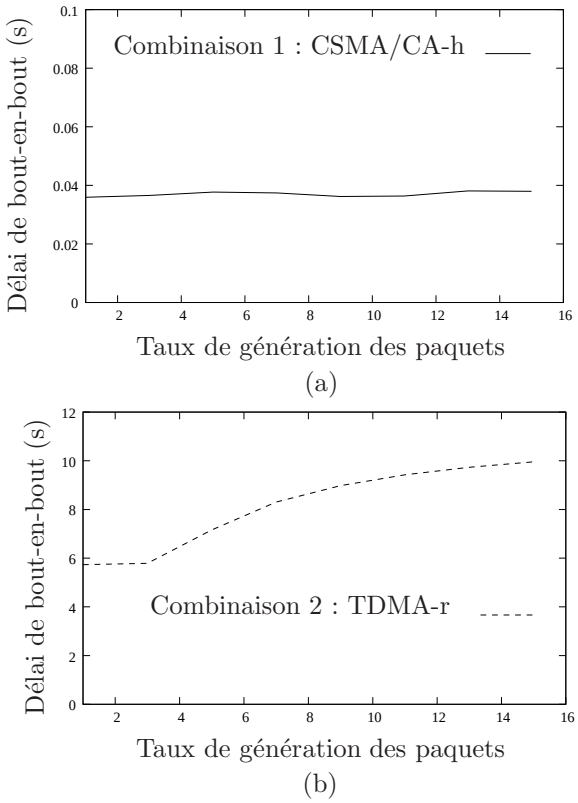


Figure 4.4 – Délai de bout-en-bout moyen en fonction du taux de génération des paquets : (a) pour la première combinaison, (b) pour la deuxième combinaison.

délais de bout-en-bout qui ne peuvent pas être tolérés pour certaines applications. En effet, une alarme doit être transmise le plus vite possible au gestionnaire du réseau afin de limiter les éventuelles conséquences.

4.2 Gestion des détours

Le problème de grand délai dans une architecture multi-couches peut être résolu en proposant un algorithme d'échange de files d'attente entre les différentes files d'attente de cette architecture (voir la partie 2.3 du chapitre 2).

Dans cette partie, nous étudions le trajet d'un paquet dans un réseau peu chargé respectant l'architecture multi-couches. Nous cherchons à identifier les risques d'apparition de détours au niveau routage en considérant un cas limite correspondant aux hypothèses simplificatrices suivantes :

– Le réseau est peu chargé : le délai d'attente dans les files d'attente est considéré nul (ce qui signifie qu'il n'y a aucune trame en instance au moment où le paquet arrive dans le nœud).

– La méthode d'accès envoie une trame en un temps constant, indépendemment de la taille de la trame. La composante aléatoire (éventuelle) de la méthode d'accès n'est pas prise en compte.
– Le médium ne génère pas de collisions et ne produit pas d'interférences.

Ces hypothèses nous permettent de faire abstraction (dans un premier temps) de la sous-couche MAC et de la couche physique, afin de centrer l'étude sur les protocoles de routage. Nous verrons plus loin que dans des conditions plus réalistes, la sous-couche MAC et la charge du réseau ont aussi un impact important sur les risques de détours (puisque la sous-couche MAC purge les trames quand le réseau est trop chargé, notamment). En pratique, dans cette partie, nous cherchons à évaluer le nombre de sauts suivi par un paquet, sans le corréler à un délai.

Dans ce qui suit, nous présentons tout d'abord les paramètres de simulations. Ensuite, nous quantifions les détours par simulation. Puis, nous présentons l'efficacité des solutions proposées dans le chapitre 2 pour éviter ces détours.

4.2.1 Paramètres de simulations

Nous générons des topologies en déployant aléatoirement des nœuds sur une surface de 100 m×100 m. Le nombre de nœuds par défaut est 100. Nous générons 1000 répétitions pour lesquelles une source aléatoire envoie un seul paquet pour une destination aléatoire. Nous calculons le nombre moyen de tentatives de sauts (ce nombre cumule le nombre de sauts et le nombre de fois où le paquet est conservé dans le nœud), de la source à la destination, produit par plusieurs combinaisons de deux protocoles de routage $\mathcal{R}_1$ et $\mathcal{R}_2$.

Les protocoles de base que nous avons utilisés sont :

1. Le protocole hiérarchique (l'arbre étant construit indépendemment pour chaque répétition avec les paramètres $C_m = 5$, $R_m = 5$ et $L_m = 10$, et le coordinateur du PAN étant choisi aléatoirement par chaque répétition).
2. Le protocole raccourci (avec les mêmes paramètres).
3. Un protocole calculant le plus court chemin (en utilisant l'algorithme de Dijkstra).
4. Le protocole OLSR.

L'instant auquel la source commence à envoyer des paquets est déterminé aléatoirement pour modéliser le fait que les paquets sont générés indépendemment de l'ordonnancement des périodes. Dans la suite, nous considérons que l'unité de temps d'une période est le temps requis

pour envoyer un paquet d'un nœud à son prochain saut, nous supposons que ce temps est constant. En pratique, ce temps dépend de la méthode d'accès et de la charge du réseau.

Nous considérons qu'un long détour a lieu dans le réseau quand un paquet n'arrive pas à atteindre sa destination finale en moins de 1000 tentatives de sauts.

4.2.2 Quantification des longs détours

Pour pouvoir quantifier les longs détours, nous nous plaçons au niveau routage. Pour montrer l'impact de la sous-couche MAC sur le nombre de longs détours, nous considérons tout d'abord une méthode d'accès idéale (mais irréaliste) qui considère que le médium est parfait (pas de collision, pas de perte). Puis, nous considérons une méthode d'accès aléatoire plus réaliste qui permet de purger des trames en cas de congestion afin d'éliminer les longs détours dans le réseau.

Médium idéal

Dans ce scénario, nous considérons que le médium est idéal : aucune trame n'est rejetée à cause de collisions ou à cause de mauvaises conditions de propagation. Toutes les trames sont de la même longueur et la méthode d'accès est déterministe. Nous considérons quatre ordonnancements de deux protocoles de routage :

1. ordonnancement 1 : le protocole de routage hiérarchique et un protocole de routage calculant le plus court chemin, représenté par h-sp sur les figures,
2. ordonnancement 2 : le protocole de routage hiérarchique et le protocole OLSR, représenté par h-OLSR sur les figures,
3. ordonnancement 3 : le protocole de routage raccourci et un protocole de routage calculant le plus court chemin, représenté par r-sp,
4. ordonnancement 4 : le protocole de routage raccourci et le protocole OLSR, représenté par r-OLSR.

Notons que seuls les protocoles de routage sont considéré dans l'ordonnancement, puisqu'ils sont les seuls à pouvoir produire des détours dans le réseau. Nous n'avons pas considérés l'ordonnancement du protocole de routage hiérarchique et du protocole de routage raccourci, ou l'ordonnancement d'un protocole de routage calculant le plus court chemin et du protocole OLSR, puisque ces protocoles sont compatibles : utilisés ensemble, ils ne peuvent pas générer de détours dans le réseau (voir partie 2.4.2 du chapitre 2).

La figure 4.5 montre le pourcentage de longs détours quand la durée des périodes varie de 1 à 5 tentatives de sauts, avec une densité de voisins égale à 9 (ce qui correspond à une portée de 20 m). Notons qu'une boucle infinie signifie qu'un paquet n'arrive jamais à sa destination finale. Quand la période est égale à 1 tentative de sauts, le protocole de routage change à chaque transmission. Quand la durée de la période augmente, la probabilité qu'un paquet entre dans une boucle infinie diminue. En effet, avec des périodes relativement grandes, les paquets ont une grande probabilité d'atteindre la destination en utilisant le même protocole de routage. Puisque le nombre moyen de tentatives de sauts produits par le pire protocole de routage (le routage hiérarchique dans notre cas) est de $7,5$, il est peu probable qu'une boucle infinie apparaisse dans le réseau si la période dure huit tentatives de sauts ou plus. Il est important de noter que même pour de grandes périodes, où l'apparition de longs détours n'est pas fréquente, les longs détours peuvent avoir un impact significatif sur le délai. En revanche, le pourcentage de longs détours est significatif pour des petites périodes. Par exemple, presque 60% des 1000 paquets générés entrent dans une boucle infinie quand l'ordonnancement 1 est considéré avec une période de durée équivalente à 1 tentative de sauts.

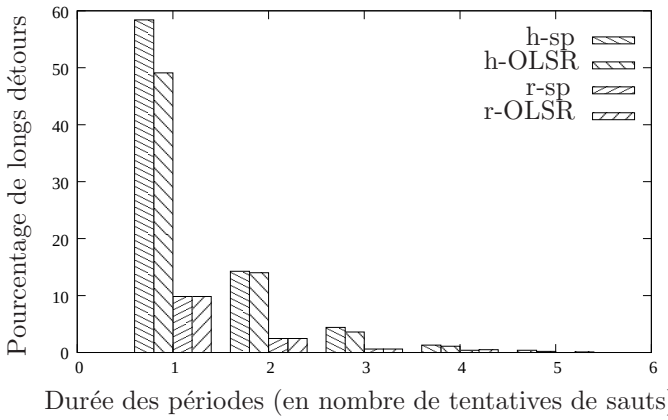


Figure 4.5 – Pourcentage de longs détours en fonction de la durée des périodes.
Le pourcentage de longs détours diminue quand la durée des périodes augmente. Dans nos topologies, le pourcentage des longs détours n'est pas significatif quand les périodes durent pour 4 sauts ou plus.

La figure 4.6 montre le pourcentage de longs détours quand les périodes ne sont pas égales. Au lieu de considérer seulement le cas où la durée de p_1 est égale à la durée de p_2 (représenté par 50%/50%), nous considérons aussi le cas où $p_1 = 2p_2$ (représenté par 67%/33%). Dans les deux cas, la durée de $p_1 + p_2$ est constante et égale à 6 tentatives de sauts. Le plus haut pourcentage de longs détours apparaît quand le pire protocole en terme de nombre de tentatives de sauts (il s'agit du protocole de routage hiérarchique ou du protocole de routage raccourci) est activé dans la plus grande période comme prévu.

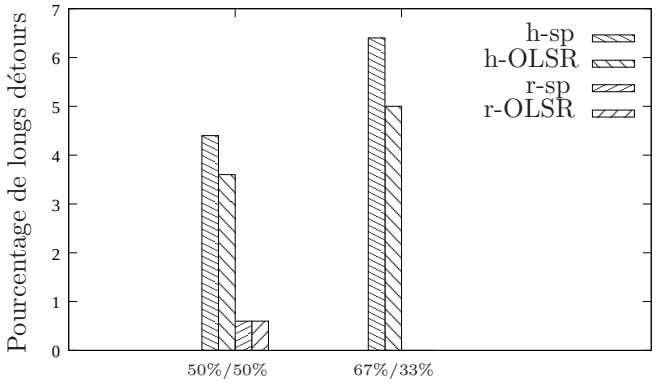


Figure 4.6 – Pourcentage de longs détours selon l'ordonnancement des périodes.
Le pourcentage de longs détours dépend de l'ordonnancement. Plus précisément, le pourcentage de longs détours augmente quand les protocoles lents sont activés pour de longues périodes.

La figure 4.7 montre le pourcentage de longs détours en fonction de la densité du réseau, pour des périodes équivalentes à trois tentatives de sauts. La densité varie de 5 voisins à 33 voisins par nœud (ce qui correspond à des portées variant de 15 m jusqu'à 40 m, avec un pas de 5 m). Le pourcentage de longs détours diminue quand la densité augmente puisque les nœuds possèdent plus d'opportunité de routage et la longueur des chemins diminue, et par conséquent les paquets sont capables d'arriver plus rapidement à la destination finale.

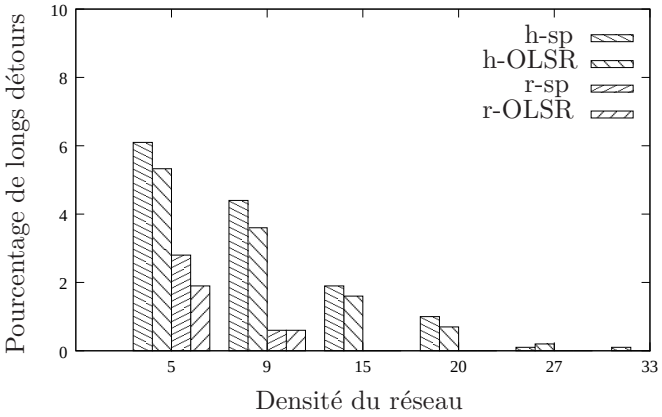


Figure 4.7 – Le pourcentage de longs détours diminue en augmentant la densité du réseau.

Médium réaliste

Nous considérons à présent que le médium est réaliste : des trames peuvent être perdues à cause de collisions ou bien à cause de mauvaises conditions de propagation. Des protocoles MAC aléatoires sont pris en compte à ce niveau. Pour modéliser l'impact des trames perdues, nous faisons varier la probabilité de transmission avec succès de la sous-couche MAC. Un taux de transmission avec une probabilité de succès de 1 correspond à un médium idéal. La durée d'une

période s'exprime à présent en terme de nombre de tentatives pour accéder au médium. Pour des raisons de simplicité, nous décidons de fixer cette durée à 5. Quand un nœud décide d'acheminer un paquet, il l'envoie à la sous-couche MAC. La sous-couche MAC informe le nœud si la trame est reçue sans faute par un de ses prochains sauts, ou bien si elle est perdue. Si la trame est perdue, elle doit être retransmise, ce qui requiert une nouvelle tentative de transmission.

Avec ce paramétrage, les longs détours sont peu probables à cause de l'aspect aléatoire dans la méthode d'accès et le médium. Par exemple, il est possible que toutes les trames soient transmises sans faute en utilisant $\mathcal{R}_1$, et que les transmissions de trames en utilisant $\mathcal{R}_2$ échouent toutes. Dans ce cas, les trames sont acheminées comme si seul $\mathcal{R}_1$ était activé dans le réseau. Même si une trame atteint le même nœud plusieurs fois, il est probable que l'aspect aléatoire de l'acheminement élimine les longs détours à terme. Cependant, le nombre de sauts parcourus par les trames peut être relativement grand. Les simulations qui suivent se concentrent davantage sur le nombre moyen de tentatives de sauts pour atteindre la destination que sur le pourcentage de longs détours. La portée des nœuds est fixée à 20 m.

La figure 4.8 illustre le nombre moyen de tentatives de sauts qu'un paquet doit parcourir avant d'arriver à sa destination finale. Ce nombre est représenté en fonction du taux de transmission avec succès de la sous-couche MAC pour plusieurs protocoles de routage. Comme prévu, le nombre de tentatives de sauts diminue quand le taux de transmission avec succès augmente. Le protocole de routage hiérarchique seul produit toujours le plus grand nombre de tentatives de sauts. Les protocoles de routage calculant le plus court chemin et le protocole OLSR produisent le plus petit nombre de sauts. Le protocole de routage raccourci conduit à un nombre de sauts intermédiaire. La figure montre de plus trois ordonnancements : ordonnancement 4, le protocole hiérarchique et le protocole raccourci, nommé par la suite ordonnancement 5 et représenté par h-r sur les figures, et le protocole calculant le plus court chemin et le protocole OLSR, nommé par la suite ordonnancement 6 et représenté par sp-OLSR sur les figures. Pour l'ordonnancement 4 et l'ordonnancement 5, le nombre moyen de tentatives de sauts est intermédiaire entre celui produit par chaque protocole considéré indépendemment. L'ordonnancement 6 montre les mêmes performances que celles d'un protocole calculant le plus court chemin : la courbe d'ordonnancement 6 est confondue avec celles du protocole calculant le plus court chemin et celles du protocole OLSR.

La figure 4.9 montre le nombre moyen de tentatives de sauts en fonction du taux de transmission avec succès de la sous-couche MAC, pour d'autres protocoles de routage. La tendance générale est similaire : le nombre de tentatives de sauts diminue quand le taux de transmission

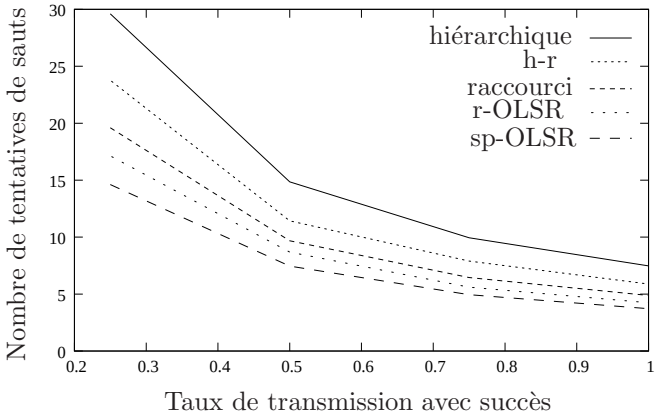


Figure 4.8 – Le nombre de tentatives de sauts pour trois ordonnancements de protocoles de routage.
Le nombre moyen de tentatives de sauts pour qu'un paquet atteigne sa destination finale est inversement proportionnel au taux de transmission avec succès de la sous-couche MAC. Pour les trois ordonnancements de protocoles de routage $\mathcal{R}_1$ et $\mathcal{R}_2$, le nombre de sauts est intermédiaire entre le nombre de sauts produits par chaque $\mathcal{R}_i$.

avec succès augmente. Dans cette figure, le nombre de tentatives de sauts produits par l'ordonnancement 1, h-sp, (respectivement par l'ordonnancement 2, h-OLSR) est beaucoup plus proche du nombre de tentatives de sauts produits par le protocole hiérarchique (qui est le protocole produisant le plus grand nombre de sauts) que le nombre de sauts produits par le protocole raccourci (respectivement OLSR) utilisé, surtout quand le taux de transmission avec succès est faible. Ce comportement diffère de celui que nous avons identifié dans la figure 4.8. L'ordonnancement 3, r-sp, a même un comportement plus défavorable : le nombre de tentatives de sauts résultant est en moyenne plus grand que le nombre de sauts produits par chaque protocole. Ceci peut être expliqué par la présence de détours (mais pas de longs détours).

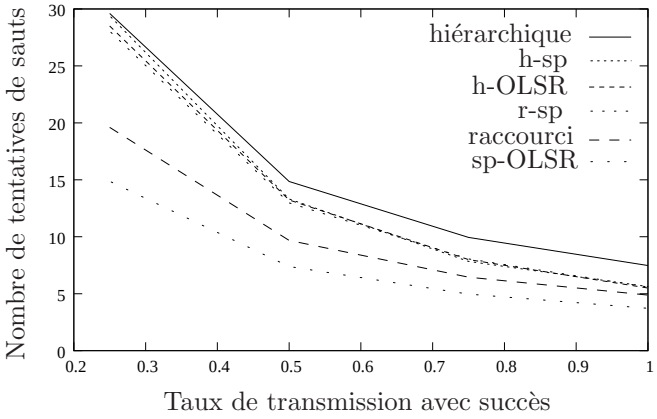


Figure 4.9 – Le nombre de tentatives de sauts pour trois autres ordonnancements de protocoles de routage.
Le nombre moyen de tentatives de sauts diminue quand le taux de transmission avec succès augmente. Pour les trois ordonnancements de protocoles de routage $\mathcal{R}_1$ et $\mathcal{R}_2$, le nombre de tentatives de sauts est proche du plus grand nombre de sauts produits par $\mathcal{R}_1$ ou $\mathcal{R}_2$, voire même plus grand.

La figure 4.10 montre le pourcentage des simulations où le nombre de tentatives de sauts produits par un scénario dépasse le nombre de sauts produits par chaque protocole de routage utilisé individuellement dans le réseau. L'ordonnancement 5, h-r, n'apparaît pas dans la figure, puisque il ne peut pas produire de détours (car les protocoles sont compatibles, voir la partie 2.4.2 du chapitre 2). L'ordonnancement 1, h-sp, et l'ordonnancement 2, h-OLSR, produisent souvent des détours. Ces détours augmentent le nombre de tentatives de sauts, ce qui explique pourquoi ces ordonnancements n'aboutissent pas à des distances intermédiaires entre les distances produites par chaque protocole à part. L'ordonnancement 3, r-sp, et l'ordonnancement 4, r-OLSR, produisent un nombre limité de détours. Ces détours apparaissent rarement, mais influent négativement sur le nombre de tentatives de sauts produits par l'ordonnancement 3 et l'ordonnancement 4.

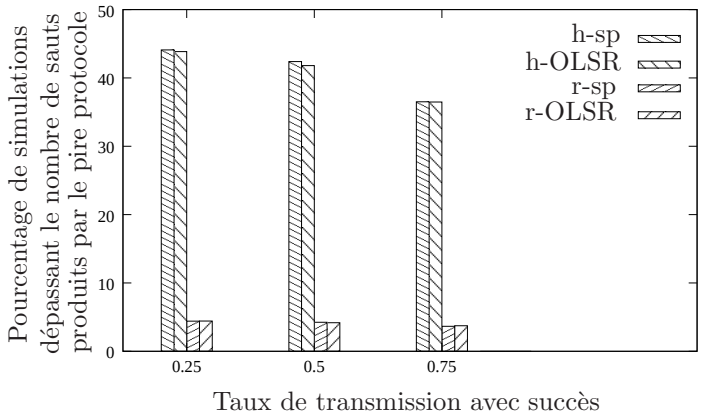


Figure 4.10 – Pourcentage de simulations dépassant le nombre de sauts produits par le pire protocole.
Pourcentage de simulations où le nombre de sauts produits par un ordonnancement dépasse le plus grand nombre de sauts produits par le pire protocole composant cet ordonnancement. Ce pourcentage est calculé en fonction du taux de transmission avec succès au niveau de la sous-couche MAC.

4.2.3 Communications intra-couche sans détours

Dans cette partie, nous montrons comment les solutions proposées dans le chapitre 2 pour éliminer les détours dans un réseau peuvent influencer sur le nombre de tentatives de sauts et sur le délai de bout-en-bout.

Protocoles de routage compatibles

La figure 4.11 montre le nombre moyen de tentatives de sauts pour arriver à la destination, en fonction de la densité du réseau, pour les protocoles compatibles. Quand le protocole de routage hiérarchique et le protocole de routage raccourci sont utilisés, ou quand un protocole de

routage calculant le plus court chemin et le protocole OLSR sont utilisés, des détours ne peuvent pas apparaître. Dans ce cas, le nombre moyen de tentatives de sauts par l'ordonnancement 5, h-r, (respectivement par l'ordonnancement 6, sp-OLSR) est intermédiaire entre le nombre moyen de tentatives de sauts correspondant au protocole hiérarchique et à celui du protocole raccourci (respectivement à celui du protocole de routage correspondant au plus court chemin et à celui du protocole OLSR). Le nombre moyen de tentatives de sauts diminue quand la densité du réseau augmente : plus la portée d'un nœud augmente, plus le chemin jusqu'à la destination est court.

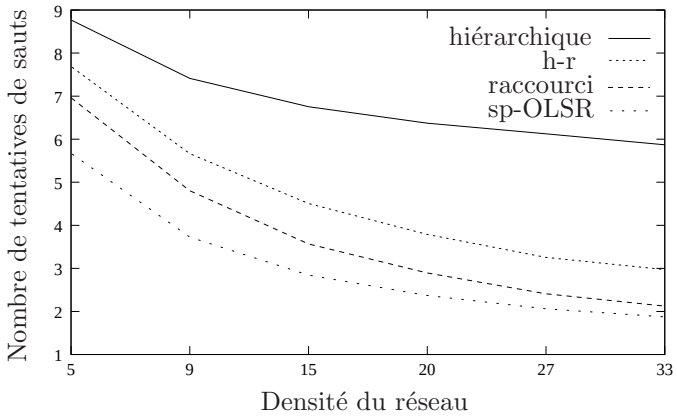


Figure 4.11 – Nombre moyen de tentatives de sauts en fonction de la densité, avec des périodes correspondant à 3 tentatives de sauts.

Protocoles de routage retardables

La figure 4.12 montre le nombre moyen de tentatives de sauts en fonction de la densité du réseau en utilisant l'approche retardable basée sur la distance du plus court chemin comme fonction de conservation. Nous considérons que le nombre de tentatives augmente de 1 à chaque fois qu'un nœud conserve le paquet. Cette approche n'aboutit pas à des détours dans le réseau, par contre, le nombre de tentatives de sauts dépasse parfois celui du pire protocole utilisé seul (le protocole de routage hiérarchique dans notre cas). Ceci est dû au fait que chaque nœud calcule la distance de son prochain saut jusqu'à la destination en utilisant la fonction de conservation et celle en utilisant le protocole de routage suivant (entre $\mathcal{R}_1$ et $\mathcal{R}_2$). Si la distance calculée par la fonction de conservation est plus petite, le nœud garde le paquet et attend l'activation du protocole de routage d'origine pour le transmettre, ce qui augmente le nombre de sauts.

La figure 4.13 montre le nombre moyen de tentatives de sauts en fonction de la densité du réseau en utilisant la distance sur l'arbre comme fonction de conservation. Puisque la fonction de conservation utilisée est la distance sur l'arbre, tous les ordonnancements possibles ne peuvent

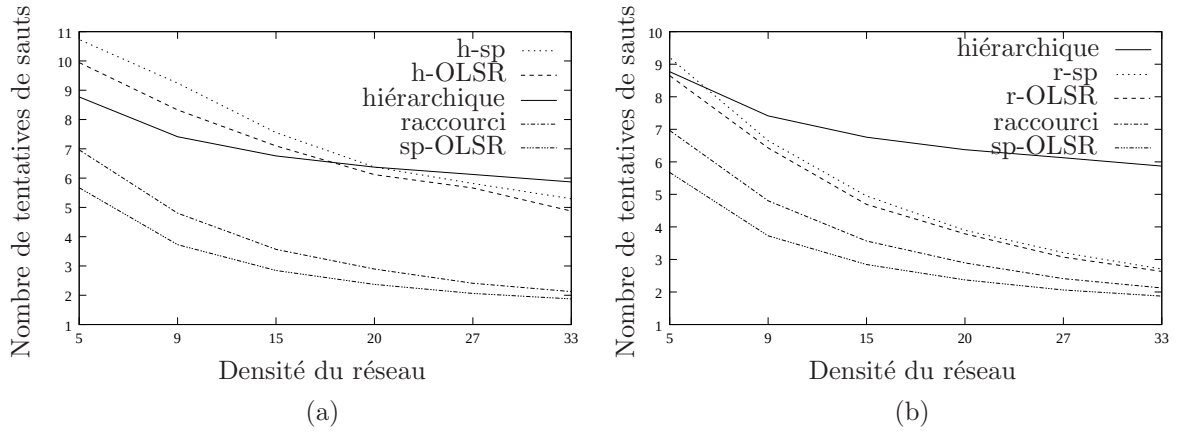


Figure 4.12 – Nombre moyen de tentatives de sauts en fonction de la densité du réseau et de la distance calculant le plus court chemin comme fonction de conservation.

pas dépasser le nombre de tentatives de sauts calculé par le pire protocole (le protocole de routage hiérarchique dans notre cas). Notons que le nombre de tentatives de sauts diminue toujours avec la densité du réseau.

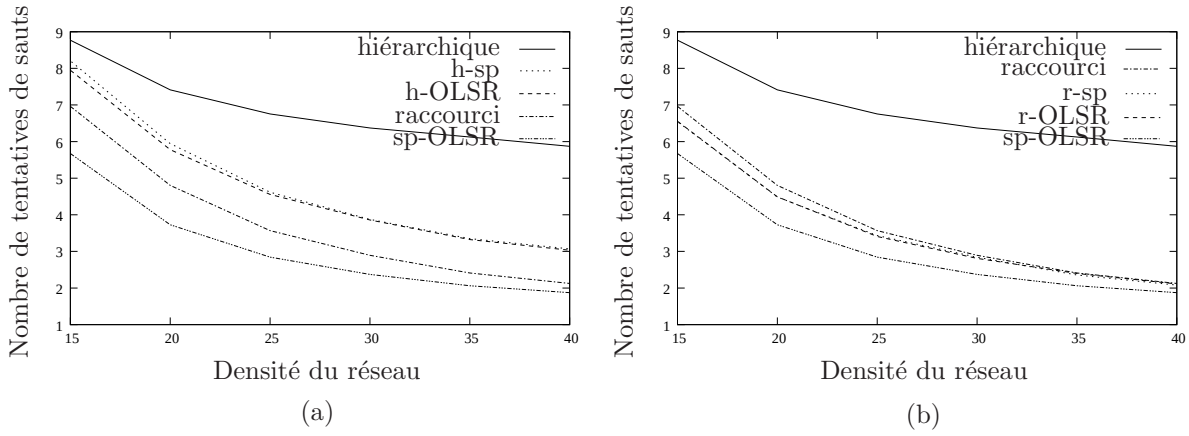


Figure 4.13 – Nombre moyen de tentatives de sauts en fonction de la densité du réseau et de la distance calculant le chemin sur l'arbre comme fonction de conservation.

Protocoles de routage combinables

La figure 4.14 et la figure 4.15 montrent le nombre moyen de tentatives de sauts selon notre approche de protocoles combinables, avec deux ensembles de protocoles et en fonction de la densité du réseau. Le protocole de routage hiérarchique est utilisé comme protocole d'attente *(stand by)* et il est combiné avec $\mathcal{R}_2$ seulement (afin d'économiser la consommation d'énergie durant la période durant laquelle $\mathcal{R}_1$ est actif (voir figure 2.15)). Aucun détour n'apparaît dans le réseau, quel que soit l'ordonnancement utilisé. Le nombre moyen de tentatives de sauts obtenu selon notre approche diminue en augmentant la densité du réseau. En moyenne, le nombre de tentatives de sauts obtenu par notre approche est intermédiaire entre le nombre de tentatives

de sauts obtenu par les deux protocoles utilisés dans l'ordonnancement. De plus, dans nos simulations, le nombre de tentatives de sauts obtenu par notre approche est toujours (et donc pas seulement en moyenne) inférieure au nombre de tentatives de sauts produits par le pire protocole parmi $\mathcal{R}_1$ et $\mathcal{R}_2$.

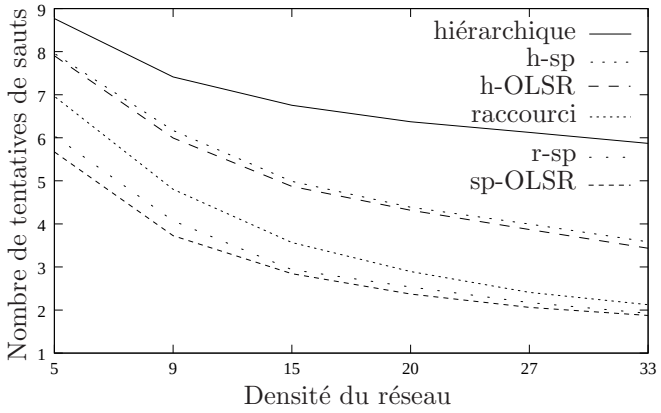


Figure 4.14 – Nombre moyen de tentatives de sauts en fonction de la densité de réseau, avec notre approche de protocoles combinables, pour des périodes équivalentes à 3 tentatives de sauts (pour le premier ensemble de protocoles).

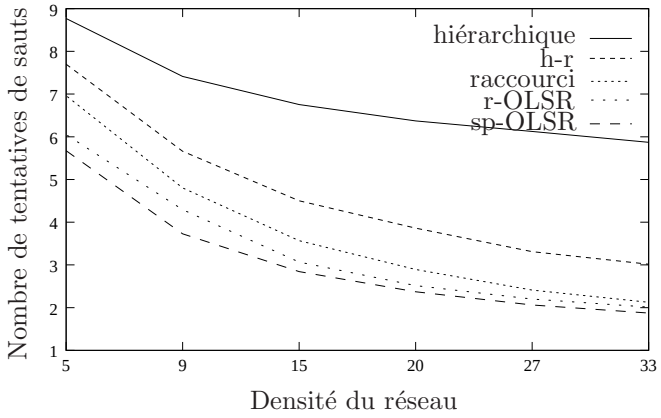


Figure 4.15 – Nombre moyen de tentatives de sauts en fonction de la densité du réseau, avec notre approche de protocoles combinables, pour des périodes équivalentes à 3 tentatives de sauts (pour le second ensemble de protocoles).

4.2.4 Bilan

Afin de fournir plusieurs QoS très différentes dans un réseau, nous avons proposé une architecture multi-couches et nous avons autorisé des changements de protocoles de routage au cours du trajet d'un paquet. Dans ces conditions, des détours peuvent toutefois apparaître. Dans cette partie, nous avons quantifié le nombre et l'impact des détours dans plusieurs scénarios. Ensuite, nous avons montré le bénéfice de plusieurs solutions aux problèmes de détours au niveau des

protocoles de routage.

4.3 Évaluation des échanges des files d'attentes dans MaCARI avec un seul type de trafic

À présent, nous considérons des hypothèses plus réalistes :

- Nous soumettons au réseau un trafic applicatif, non négligeable, ce qui nous permet de considérer des réseaux dont la charge varie. Ceci met en évidence le rôle des files d'attente.
- Nous utilisons une méthode d'accès complète, intégrant une gestion de files d'attente de taille limitée et une gestion du temps appropriée.
- Nous modélisons un médium réaliste qui suit le modèle ITU (et gérant donc les collisions et les interférences).

Nous utilisons la méthode d'accès MaCARI pour montrer les performances de notre approche hybride. Plus précisément, nous utilisons la période $[T_2; T_3]$ de MaCARI comme une période de secours (servant de soupape) pour soulager la période $[T_1; T_2]$. Tout d'abord, nous décrivons la topologie que nous avons testée et le modèle de trafic simulé. Ensuite, nous détaillons les paramètres de simulations. Finalement, nous présentons les résultats de simulations de notre approche en considérant deux scénarios : le premier où le cycle global dure une seconde et contient une période d'inactivité, et le second où la durée du cycle global est inférieure à une seconde et ne contient pas une période d'inactivité. Les métriques étudiées sont le débit et le délai de bout-en-bout.

4.3.1 Paramètres de simulations

Les simulations sont lancées en utilisant NS2. Nous avons gardé la même grandeur de réseau (une cinquentaine de nœuds). Nous avons choisi de considérer une topologie industrielle réelle proposée par un des partenaires du projet ANR OCARI, afin d'évaluer les performances de notre approche. Une représentation logique de cette topologie est illustrée sur la figure 4.16. Sur cette figure, seuls les coordinateurs sont représentés : les feuilles ne sont pas montrées (mais chaque coordinateur a au moins 2 feuilles). Le coordinateur du PAN est représenté par un double cercle et les coordinateurs par des cercles. Les liens solides représentent les associations père-fils alors que les pointillés représentent les liens radio opérationnels. La faible densité de nœuds est due à la présence de murs.

Les feuilles produisent des trames qui sont collectées par leurs coordinateurs. En moyenne,

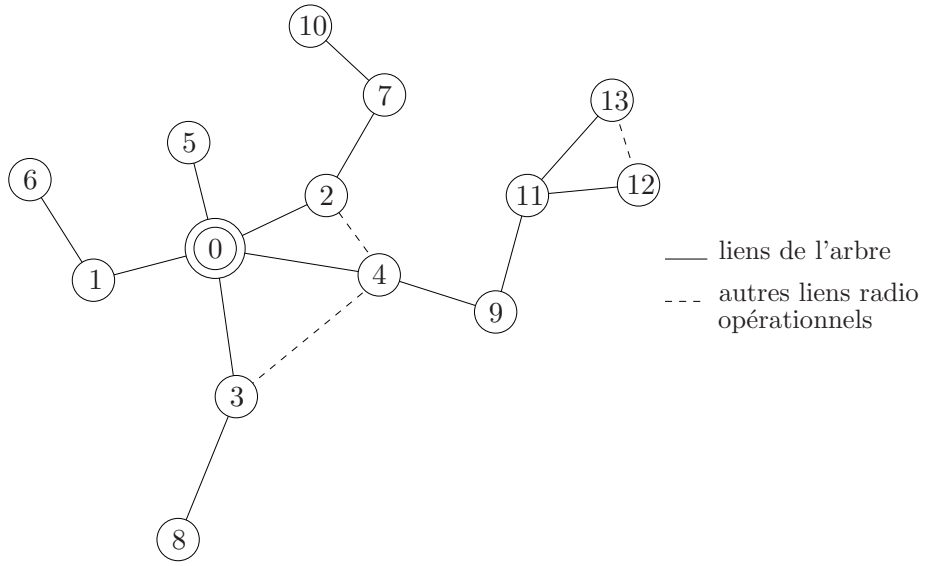


Figure 4.16 – Topologie d'un réseau de capteurs sans fil industriel.

chaque coordinateur collecte de ses feuilles 5 trames chaque 2 secondes. La destination du trafic est le coordinateur du PAN. Puisque les coordinateurs acheminent les trames de leurs fils, ceux qui sont normalement proches du coordinateur du PAN sont surchargés. Par exemple, il est probable que les coordinateurs 2, 4 et 9 aient des trames perdues.

Puisque notre topologie représente l'intérieur d'un bâtiment, nous avons utilisé le modèle de propagation en intérieur de l'ITU, ITU-R P1238-4 [ITU05], qui introduit un aspect probabiliste sur l'état des liens.

Durant le cycle global, la durée totale de la période de synchronisation pour les 14 coordinateurs est d'environ 40 ms. Nous réservons 20 ms à chaque coordinateur pour l'activité intra-étoile. Cette durée est suffisante pour que le coordinateur de l'étoile puisse collecter entre 2 ou 3 trames en utilisant le mécanisme CSMA/CA slotté [CLG+09].

Nous faisons varier la durée des intervalles TDMA de 30 ms à 10 ms. Quand l'intervalle TDMA est de 30 ms, il n'y a pas de période CSMA/CA. Quand l'intervalle de temps est égal à 20 ms ou bien à 10 ms, il y a une période CSMA/CA de durée constante égale à 70 ms. Quand le cycle global est fixe, une période d'inactivité variable $[T_3; T_0]$ est utilisée. Le paramétrage du cycle global constant est illustré dans la figure 4.17. Quand les intervalles de temps TDMA durent 20 ms, la période $[T_1; T_3]$ dure 70 ms de moins que quand les intervalles de temps TDMA durent 30 ms. Quand les intervalles de temps TDMA durent 10 ms, la période $[T_1; T_3]$ dure 210 ms de moins que quand les intervalles de temps durent 30 ms. À mesure que les intervalles TDMA diminuent, la durée de la période d'activité diminue aussi. Le paramétrage du cycle

global variable est semblable, sans la période d'inactivité $[T_3; T_0]$.

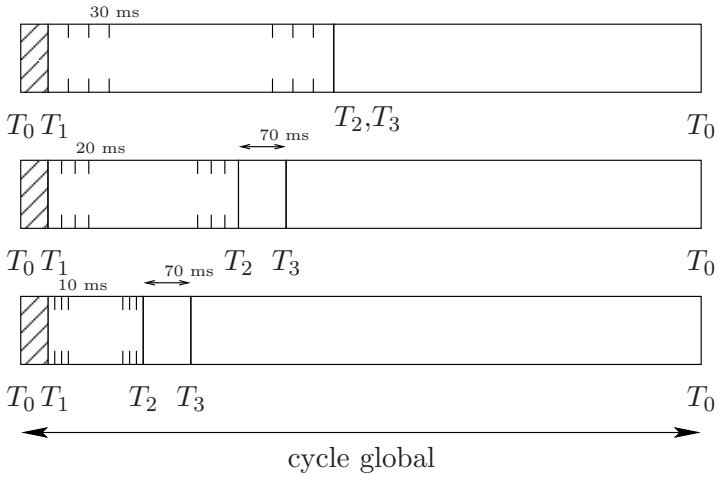


Figure 4.17 – Paramétrage du cycle global constant.

Nous définissons le cycle d'activité comme le pourcentage de temps d'un cycle global durant lequel une entité est active. Quand le cycle global est de 1 seconde, le cycle d'activité pour un coordinateur ayant 5 fils, comme le coordinateur du PAN est de 24%, 25% ou 19%, pour respectivement un intervalle de temps TDMA égal à 30 ms, 20 ms ou 10 ms. Le cycle d'activité pour les feuilles est toujours de 6%. Quand le cycle global n'inclut pas une période d'inactivité $[T_3; T_0]$, le cycle d'activité pour un coordinateur ayant 5 fils est respectivement de 30%, 37% ou 36% pour un intervalle de temps TDMA égal à 30 ms, 20 ms ou 10 ms respectivement. Le cycle d'activité des feuilles est respectivement 7%, 9% ou 11%, pour un intervalle de temps TDMA de 30 ms, 20 ms ou 10 ms.

La simulation est lancée pour 40 secondes. En outre, nous avons décidé de limiter la taille des files d'attente à 20 trames de 23 octets (au niveau MAC).

4.3.2 Débit

Le débit est défini par le nombre de trames reçues par la destination finale (qui est le coordinateur du PAN dans nos simulations) par unité de temps. Le débit est représenté dans les schémas qui suivent en fonction de la durée des intervalles de temps TDMA.

La figure 4.18 montre le débit pour un cycle global constant fixé à 1 seconde pour l'approche TDMA seul existante et pour notre approche représentée par hybride sur la figure. Nous remarquons qu'en réduisant la durée des intervalles de temps TDMA, le débit total augmente. Quand les intervalles TDMA sont de 30 ms, notre approche hybride n'est pas exécutée. En effet, toutes les trames stockées dans la file d'attente peuvent être traitées durant les intervalles TDMA. Les

bénéfices de notre approche hybride sont montrés en réduisant la durée des intervalles TDMA de 30 ms à 20 ms ou à 10 ms. Dans ces cas, le gain produit par notre approche est de 3% pour des intervalles TDMA de 20 ms et atteint 26% pour une des intervalles TDMA de 10 ms. Ce gain est dû au fait qu'en utilisant la période CSMA/CA pour absorber l'accumulation du trafic, les trames ont plus d'opportunités d'être transmises, ce qui augmente le débit dans le réseau. Notre approche hybride répond à la contrainte d'économie d'énergie qui consiste à avoir la plus grande période d'inactivité $[T_3; T_0]$ tout en préservant le débit. Si l'on se compare au nombre de paquets envoyés et au temps d'activité quand les intervalles TDMA ont une durée de 30 ms, notre approche hybride offre les performances suivantes : elle permet d'envoyer 90% des paquets en étant active seulement 83% du temps, quand les intervalles TDMA durent 20 ms, et elle permet d'envoyer 78% des paquets en étant active seulement 50% du temps, quand les intervalles TDMA durent 10 ms.

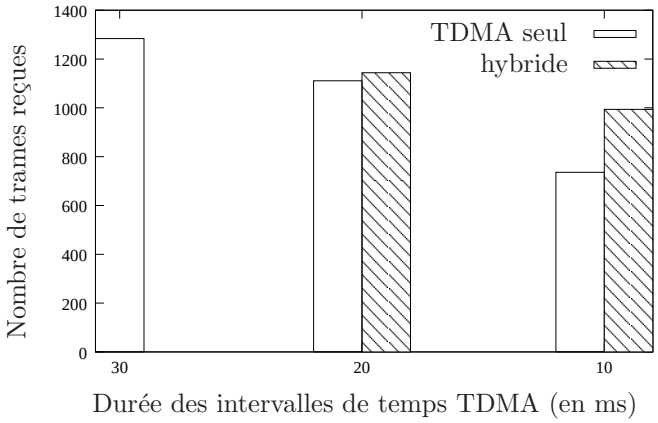


Figure 4.18 – Débit pour un cycle global avec une période d'inactivité.

La figure 4.19 montre le débit quand nous considérons un cycle global de durée variable. Dans ce scénario, il n'y a pas de période d'inactivité $[T_3; T_0]$. Le cycle global dure toujours moins d'une seconde. Les avantages de notre approche hybride peuvent être identifiés quand les intervalles de temps TDMA durent pour 10 ms : 13% de trames supplémentaires sont envoyées avec notre approche par rapport à l'approche TDMA seul.

4.3.3 Délai de bout-en-bout

Le délai de bout-en-bout est défini comme la durée entre la transmission d'une trame par la source et sa réception par la destination finale. Le délai de bout-en-bout ne prend en compte que les trames qui sont correctement reçues. Dans nos résultats, nous montrons seulement la distribution du délai de bout-en-bout pour des intervalles de temps TDMA de 10 ms, puisque

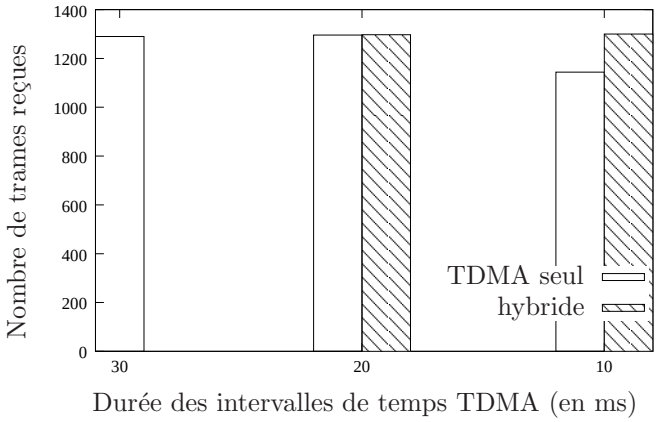


Figure 4.19 – Débit pour un cycle global sans période d'inactivité.

c'est pour cette valeur que notre approche hybride est la plus avantageuse.

La figure 4.20 montre la distribution du délai de bout-en-bout pour un cycle global d'une seconde. Les trames ont un délai de bout-en-bout maximal de 8 secondes en utilisant l'approche TDMA seul et seulement 7 secondes en utilisant l'approche hybride. Nous remarquons que le délai de bout-en-bout est en moyenne plus petit avec la méthode hybride qu'avec TDMA seul. En moyenne, le délai est de $1,85$ secondes avec l'approche hybride, alors qu'il est d'environ 3 secondes pour l'approche TDMA seul. Ceci correspond à un gain de 39%. Avec TDMA seul, 140 trames sont reçues avec un délai de bout-en-bout inférieur à une seconde. Avec l'approche hybride, 608 trames sont reçues avec le même délai (c'est-à-dire $4,3$ fois plus). Ceci est dû au fait que notre approche permet aux trames d'être envoyées durant la période CSMA/CA au lieu d'attendre l'occurrence de leur prochain intervalle TDMA.

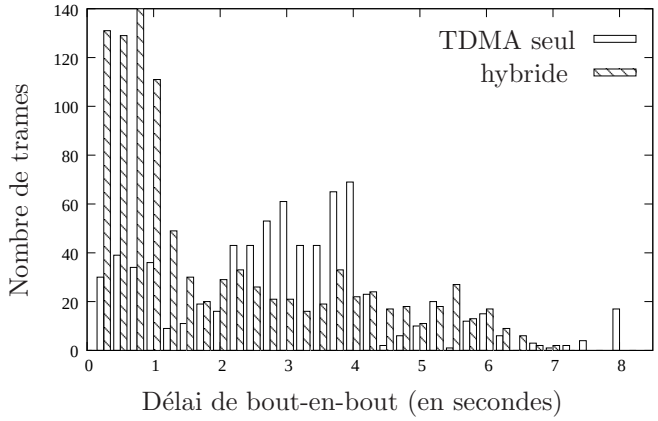


Figure 4.20 – Distribution du délai de bout-en-bout pour un cycle global avec une période d'inactivité.

La figure 4.21 illustre la distribution du délai de bout-en-bout pour un cycle global qui ne contient pas la période d'inactivité $[T_3; T_0]$. À mesure que la durée des intervalles TDMA est

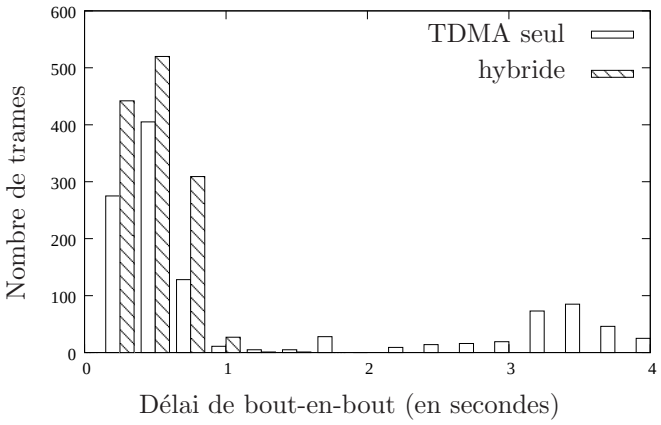


Figure 4.21 – Distribution du délai de bout-en-bout pour un cycle global sans une période d'inactivité.

réduite, le cycle global diminue, ce qui réduit le délai de bout-en-bout pour les deux approches : le délai de bout-en-bout maximal pour l'approche TDMA seul est réduit de 8 s à 4 s (comparé à la figure 4.20) et le délai maximal pour notre approche hybride est réduit de 7 s à $1,5$ s. Nous remarquons que le délai de bout-en-bout moyen est d'environ $1,1$ secondes pour TDMA seul et de moins de $0,4$ seconde pour hybride, ce qui correspond à un gain de 64%. En outre, presque 700 trames sont reçues en moins d'une seconde en utilisant TDMA seul et une partie significative des trames attendent plusieurs cycles pour arriver à la destination. Par contre, 1300 trames sont reçues en utilisant hybride pour la même durée de temps et la majorité des trames arrivent à la destination dans moins de 2 cycles. L'impact fort de la période $[T_3; T_0]$ sur les délais de bout-en-bout est confirmé ici.

Notons que l'approche hybride améliore le débit ainsi que le délai de bout-en-bout pour les deux scénarios : cycle fixe à une seconde et cycle de durée inférieure à une seconde et sans période d'inactivité.

4.3.4 Bilan

Dans cette partie, nous avons évalué l'approche hybride permettant de réduire l'accumulation du trafic due à un dimensionnement statique du mécanisme TDMA. Notre approche consiste à utiliser, après les intervalles TDMA, la période CSMA/CA de MaCARI qui est commune à tous les coordinateurs du réseau pour traiter le surplus de trafic de la période TDMA. Dans nos simulations, nous avons considéré un cycle global constant avec une période d'inactivité puis sans période d'inactivité. Les résultats de simulations montrent que notre approche améliore l'approche traditionnelle du mécanisme TDMA en termes de débit (pour un intervalle

TDMA de 10 ms, la réduction du débit est de 26% pour un cycle fixe et de 13% pour un cycle sans période d'inactivité) et de délai de bout-en-bout (pour un intervalle de 10 ms, la réduction du délai de bout-en-bout moyen est de 39% pour un cycle fixe et de 64% pour un cycle sans période d'inactivité). Nous avons montré aussi que la période d'inactivité $[T_3; T_0]$ impacte les délais de bout-en-bout moyen.

4.4 Évaluation des échanges de files d'attente dans MaCARI avec deux types de trafic

Dans cette partie, nous réalisons des simulations afin de quantifier les bénéfices de notre mécanisme d'échanges de files d'attente. Tout d'abord, nous décrivons les paramètres de simulation. Ensuite, nous présentons les résultats de simulations en considérant trois scénarios : sans échange de files d'attente, avec échanges pour le trafic non contraint, nommé trafic #2, seulement (c'est-à-dire que les trames de $[T_2; T_3]$ peuvent être envoyées aussi durant $[T_1; T_2]$), et avec échanges de files d'attente pour les deux trafic. Les métriques considérées sont le débit, le taux de pertes et le délai de bout-en-bout.

4.4.1 Paramètres de simulations

Les simulations sont toujours réalisées en utilisant le simulateur réseau NS2. Nous considérons un ensemble de 40 coordinateurs aléatoirement distribués dans une surface de 30 m×30 m. Nous utilisons le modèle de propagation en intérieur de l'ITU. Le coordinateur du PAN est localisé au centre de la surface. Les paramètres du réseau sont fixés à : $L_m = 6$, $R_m = 5$ et $C_m = 4$. Le cycle global est égal à 4 secondes. La durée de la période de synchronisation $[T_0; T_1]$ pour les 40 coordinateurs est 256 ms (chaque balise prend environ 20 périodes de *backoff* de 320 μs pour être envoyée par un coordinateur, puisque les balises de MaCARI sont des grandes trames). La durée des deux périodes $[T_1; T_2]$ et $[T_2; T_3]$ est chacune égale à $1,28$ secondes. Le reste du cycle global est consacré à la période d'inactivité $[T_3; T_0]$. Tous les coordinateurs génèrent du trafic à destination du coordinateur du PAN. Chaque source génère 1 trame de 30 octets (au niveau physique) chaque 2 secondes, et la marque avec #1 (pour le trafic contraint) ou #2 (pour le trafic non contraint) selon la proportion des deux trafics.

Afin d'évaluer notre mécanisme, nous calculons les métriques une fois que le réseau a atteint un état stationnaire. Nous observons donc l'évolution des performances du réseau à partir de la $30^{\text{ème}}$ seconde et pour une durée de 60 secondes.

4.4.2 Débit

Le débit est défini comme le nombre de trames reçues par la destination finale (dans nos simulation, la destination finale est toujours le coordinateur du PAN) par unité de temps. Le débit est représenté en fonction du ratio du trafic #1 sur le cumul de trafics (trafic #1+trafic #2). Par exemple, le ratio 2/8 signifie que 2 trames générées sur 8 correspondent au trafic #1 (et sont donc reliées à la période $[T_1; T_2]$), et que 6 trames correspondent au trafic #2 (et sont donc reliées à la période $[T_2; T_3]$).

La figure 4.22 montre le débit pour les trois scénarios : (1) sans appliquer notre mécanisme, et donc sans échange de paquets entre les files d'attente, (2) en appliquant le mécanisme dans un seul sens (c'est-à-dire en autorisant l'envoi de trames de la période $[T_2; T_3]$ pendant la période $[T_1; T_2]$) et (3) en appliquant le mécanisme dans les deux sens (c'est-à-dire que les trames peuvent être envoyées dans les deux périodes, quel que soit leur marquage).

Pour le scénario 1, nous remarquons que le débit augmente jusqu'à ce que le ratio de trafic #1 sur le cumul de trafics atteigne la valeur 2/8, puis il diminue. Avec un ratio de 1/8, les intervalles de relais de $[T_1; T_2]$ sont assez grands pour le trafic : le temps est gaspillé durant $[T_1; T_2]$. Avec un ratio de 2/8, les intervalles de relais de $[T_1; T_2]$ sont bien dimensionnés pour le trafic. Avec les ratios 3/8 et 4/8, les intervalles de relais de $[T_1; T_2]$ sont trop petits pour le trafic généré. Dans ce cas, certaines trames sont perdues durant $[T_1; T_2]$: la performance globale en terme du débit diminue.

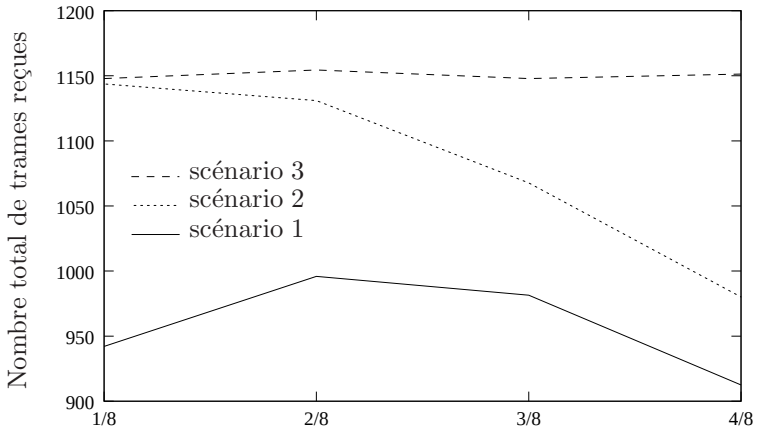


Figure 4.22 – Débit moyen du réseau.

Pour le scénario 2, le débit est toujours plus élevé que pour le scénario 1. Quand la période $[T_2; T_3]$ est surchargée (c'est le cas pour des petits ratios), la période $[T_1; T_2]$ n'est pas surchargée : la file d'attente $\mathcal{Q}_1$ du trafic #1 est souvent vide et des échanges peuvent avoir lieu, ce qui réduit

la charge de $[T_2; T_3]$ et améliore les performances globales du réseau. Quand la période $[T_1; T_2]$ est surchargée (c'est le cas pour des ratios élevés), la file d'attente $\mathcal{Q}_1$ du trafic #1 est rarement vide et peu d'échanges peuvent avoir lieu : les performances se rapprochent de celles du scénario 1. En résumé, notre mécanisme d'échanges entre les files d'attente permet d'améliorer très significativement les performances du réseau en termes de débit.

Pour le scénario 3, le débit est constant. Il n'y a pas de pertes de temps dans les deux périodes : $[T_1; T_2]$ et $[T_2; T_3]$ sont utilisées pour envoyer des trames du trafic #1 et du trafic #2. Le gain entre le scénario 1 et le scénario 3 varie entre 15% et 21% selon le ratio du trafic #1 sur le trafic #2.

4.4.3 Taux de pertes

Le taux de pertes est défini comme le ratio du nombre de trames correctement reçues par la destination finale sur le nombre de trames générées par les nœuds sources. Cette définition considère que les trames en file d'attente à la fin de la simulation sont perdues. Toutefois, le nombre de trames correctement reçues intègre les trames qui étaient en file d'attente au moment où les métriques sont calculées (c'est-à-dire à partir de la $30^{\text{ème}}$ seconde). Comme notre hypothèse est que le comportement du réseau est stationnaire, nous supposons que le nombre de trames dans la file d'attente à la $30^{\text{ème}}$ seconde est égal au nombre de trames dans la file d'attente à la $90^{\text{ème}}$ seconde.

La figure 4.23 montre le taux de pertes moyen pour le trafic #1 en fonction du ratio du trafic #1 sur le cumul de trafics. Pour le scénario 1 et le scénario 2, le taux de pertes du trafic #1 augmente avec le ratio. Cette augmentation est due au fait que les intervalles de relais de $[T_1; T_2]$ et la taille de la file d'attente $\mathcal{Q}_1$ ne sont pas bien adaptés à un grand nombre de trames. Il y a peu de différence entre le scénario 1 et le scénario 2 puisque l'échange concerne seulement le trafic #2 (dans le scénario 2). Pour le scénario 3, le pourcentage de trames perdues reste minime puisque le trafic total est constant et peut être globalement envoyé grâce à la banalisation des deux périodes.

La figure 4.24 montre le taux de pertes moyen pour le trafic #2 en fonction du ratio du trafic #1 sur le cumul de trafics. Pour le scénario 1, le taux de pertes des trames du trafic #2 diminue quand le ratio augmente. Avec un petit ratio, plusieurs trames sont générées pour le trafic #2, et les nœuds souffrent d'une contention élevée pour accéder au médium. Avec un grand ratio, moins de trames sont générées pour le trafic #2, ce qui réduit la contention pour le médium. Pour le scénario 2 et le scénario 3, les deux périodes peuvent être utilisées pour le

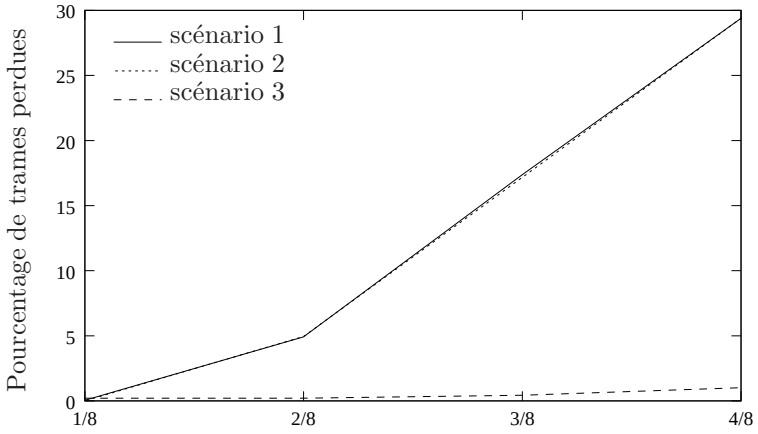


Figure 4.23 – Taux de pertes moyen (en pourcentage) pour le trafic #1.

trafic #2 et le trafic global est constant, ce qui aboutit à un taux de pertes constant (et petit).

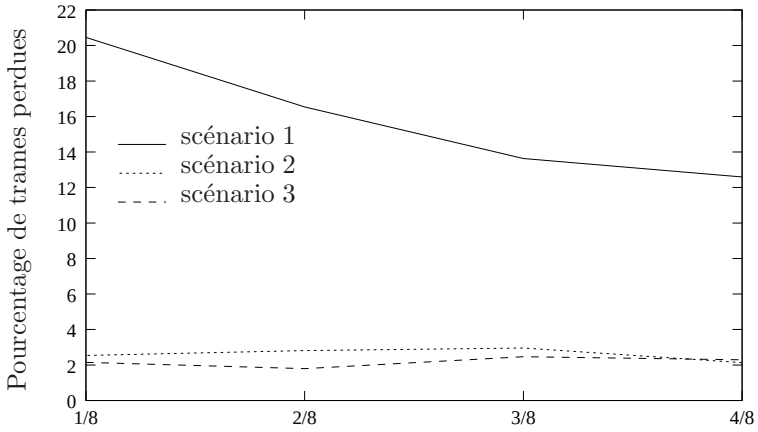


Figure 4.24 – Taux de pertes moyen (en pourcentage) pour le trafic #2.

4.4.4 Délai de bout-en-bout

La figure 4.25 montre la distribution du délai de bout-en-bout pour le trafic #1. L'axe des ordonnées est normalisé : il est représenté comme un pourcentage des trames (plutôt qu'un nombre de trames). Le délai de bout-en-bout maximal est de 9 secondes pour le scénario 1, 8,5 secondes pour le scénario 2, et 4,5 secondes pour le scénario 3. Pour le scénario 1, seulement 2% des trames sont reçues dans un délai de moins d'une seconde. En effet, avec le scénario 1, les trames souffrent d'un grand délai parce qu'elles ne sont pas envoyées durant $[T_2; T_3]$. Pour le scénario 2, les résultats sont similaires puisque les trames du trafic #1 ne peuvent pas être envoyées durant $[T_2; T_3]$. Pour le scénario 3, 13% des trames sont acheminées en moins d'une seconde. Globalement, le délai de bout-en-bout moyen pour le scénario 3 (qui est de 2,247

secondes) représente seulement 52% du délai calculé dans le scénario 1 et le scénario 2 (qui est de $4,257$ secondes).

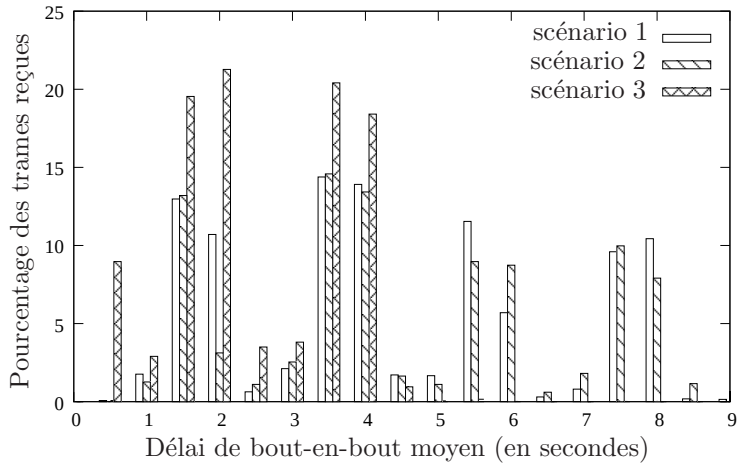


Figure 4.25 – Distribution du délai de bout-en-bout moyen pour le trafic #1.

La figure 4.26 montre la distribution du délai de bout-en-bout moyen pour le trafic #2. L'axe des ordonnées est là aussi normalisé. Le délai de bout-en-bout maximal des trames est de 6 secondes pour le scénario 1, et $2,5$ secondes pour le scénario 2 et le scénario 3, ce qui représente un gain de 58%. Les trames du trafic #2 ont un délai plus petit que celles du trafic #1 parce que les trames du trafic #2 peuvent être envoyées durant les deux périodes $[T_1; T_2]$ et $[T_2; T_3]$. De plus, les chemins vers la destination calculés durant $[T_2; T_3]$ sont plus courts que ceux calculés durant $[T_1; T_2]$.

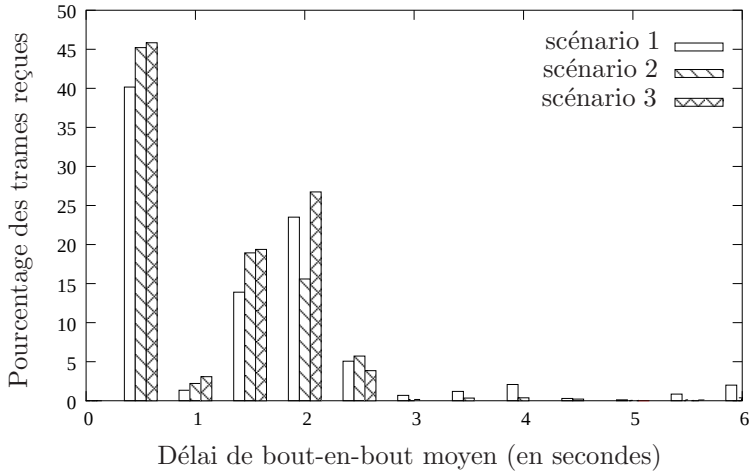


Figure 4.26 – Distribution du délai de bout-en-bout moyen pour le trafic #2.

4.4.5 Bilan

Dans cette partie, nous avons montré l'intérêt des échanges de files d'attente dans un réseau multi-couches. Les résultats de simulations qui ont permis de prendre en compte le véritable rôle des files d'attente des nœuds, ont montré qu'avec notre mécanisme d'échanges, le problème de dimensionnement des périodes des architectures multi-couches n'est plus critique. De plus, avec notre mécanisme, les métriques du réseau sont améliorées. Pour quantifier ceci, nous avons utilisé la méthode d'accès MaCARI sur une configuration représentative et avec un processus de simulation prenant en compte l'aspect aléatoire du médium. Nous avons pu augmenter le débit de 21% et réduire le délai de bout-en-bout de 52% en appliquant notre mécanisme.

4.5 Synthèse

Dans ce chapitre, nous avons présenté les différents résultats reliés à l'architecture multi-couches que nous avons proposée dans le chapitre 2. Les architectures multi-couches assurent la QoS pour plusieurs applications qui peuvent être déployées dans un même réseau. Les résultats ont montré qu'il est intéressant d'avoir plusieurs combinaisons de protocoles MAC et réseau dans un réseau. Une combinaison peut être convenable pour la transmission rapide de données mais ne garantit pas la réception des données, alors qu'une autre combinaison peut assurer la bonne réception des données mais avec un grand délai de bout-en-bout. Cependant, ces architectures multi-couches ont un problème de dimensionnement des périodes allouées pour chaque combinaison. Un autre problème que posent ces architectures est le délai.

La solution que nous avons étudiée consiste à autoriser les échanges de paquets entre les combinaisons $(\mathcal{M}_i, \mathcal{R}_i)$. Ces échanges risquent d'aboutir à des détours dans le réseau, si l'on considère des échanges entre des protocoles de routage arbitraires. Pour évaluer ces risques de détours, nous avons considéré initialement le trajet d'un paquet, en simplifiant le comportement de la méthode d'accès et du médium, et en considérant un réseau très peu chargé. Après avoir fait ces hypothèses, nous avons étudié l'apparition de détours sur plusieurs protocoles de routage, et nous avons testé les solutions que nous avons proposées dans le chapitre 2 pour les éviter. Au final, nous avons conclu que certains protocoles de routage peuvent coexister étant donné qu'ils sont compatibles entre eux. Cette conclusion reste valide avec des hypothèses plus réalistes.

Une fois le problème de détours réglé en considérant des combinaisons de protocoles compatibles, nous avons implémenté le mécanisme d'échanges de files d'attente dans une architecture multi-couches. Pour cela, nous avons intégré le mécanisme d'échanges au protocole MaCARI,

qui possède des protocoles de routage compatibles (protocole de routage hiérarchique durant la période $[T_1; T_2]$ et une version optimisée du protocole de routage hiérarchique durant la période $[T_2; T_3]$). Nous avons, tout d'abord, considéré un seul type de trafic que nous avons généré dans la période $[T_1; T_2]$, et nous avons augmenté la charge jusqu'à ce que cette période ne puisse plus supporter le trafic. Nous avons utilisé la période $[T_2; T_3]$ comme une période de secours pour absorber l'accumulation du trafic de $[T_1; T_2]$. Avec cette stratégie, nous avons montré que nous pouvons augmenter le débit et réduire le délai. Ensuite, nous avons testé les performances du réseau avec plusieurs types de trafic. Pour cela, nous avons testé le mécanisme d'échanges en supposant deux types de trafic. Les résultats ont montré qu'avec notre mécanisme, les performances du réseau sont significativement améliorées.

Chapitre 5

Contributions complémentaires

La mise en œuvre et l'exploitation d'un réseau de capteurs a lieu en deux phases : une phase de déploiement initial (et de configuration) et une phase de communications. Durant la phase d'installation, le réseau s'organise afin de pouvoir détecter tous les nœuds. Durant la phase de communications, des données sont échangées entre les nœuds.

Dans ce chapitre, nous décrivons deux contributions complémentaires à nos travaux sur les échanges de files d'attente. La première contribution complémentaire identifie un problème de durée de déploiement de grands réseaux de capteurs dû à la congestion et le résout en proposant des mécanismes permettant d'associer un grand nombre de nœuds à un réseau. La deuxième contribution complémentaire consiste à réduire la congestion durant la phase de communications, ce qui permet de diminuer le délai et le taux de pertes. Dans cette contribution, nous proposons des protocoles de routage basés sur des pivots.

5.1 Congestion lors du déploiement du réseau

La phase de déploiement d'un réseau, aussi appelée la phase d'installation, est la phase durant laquelle les messages de contrôle, de détection et de configuration sont échangés. Ces messages permettent d'aboutir à un réseau complètement opérationnel.

Durant cette phase, des problèmes de congestion dûs au grand nombre de messages de contrôle peuvent empêcher les nœuds de se connecter rapidement. Ces problèmes augmentent la durée d'installation du réseau et retardent la phase de communications.

Cette partie s'attaque au problème de congestion durant la phase d'installation du réseau.

Pour cela, nous détaillons les problèmes qui peuvent apparaître durant cette phase, et nous proposons trois mécanismes qui contribuent à leur réduction.

5.1.1 Phase d'association

La durée d'installation d'un réseau *(setup time)* est définie par la durée de temps requise pour que ce réseau devienne opérationnel. En d'autres termes, il s'agit de l'intervalle de temps durant lequel tous les nœuds s'organisent. Le temps d'installation est généralement assez long, mais souvent négligé puisqu'il n'a lieu qu'une seule fois (sauf modification de topologie), durant le déploiement du réseau.

Cependant, un temps d'installation long cause des problèmes dans plusieurs applications. Par exemple, quand un réseau doit être déployé rapidement pour surveiller des situations critiques (après un effondrement d'un bâtiment ou durant une mission militaire), un temps d'installation court est crucial. De même, quand tous les composants d'un réseau redémarrent simultanément (suite à une panne), la récupération rapide du réseau dépend du temps d'installation. Un autre exemple est quand un nœud tombe en panne et que cela déconnecte une partie du réseau. Tous les nœuds déconnectés essaient de se reconnecter en même temps, ce qui cause des problèmes de congestion et par conséquent de larges délais.

Dans la suite de cette partie, nous nous rapprochons de la terminologie IEEE 802.15.4/ZigBee en appelant le temps d'association des nœuds le temps d'installation du réseau.

5.1.2 Mécanisme d'affectation d'adresses

Une tâche importante de la phase d'installation d'un réseau est l'affectation d'adresses (c'est-à-dire l'affectation d'une adresse unique à chaque nœud). Dans cette partie, nous étudions les problèmes d'affectation d'adresses qui peuvent survenir dans un réseau. Nous prenons pour exemple un réseau déployé dans une mine, où la détection de fuites de gaz est critique pour la sécurité des humains y travaillant. Les réseaux de capteurs sont adaptés pour un tel environnement, alors que les réseaux filaires nécessitent l'installation des câbles.

Dans la suite de cette partie, nous détaillons les problèmes d'associations et d'affectation d'adresses qui aboutissent à des problèmes de congestion durant la phase de déploiement d'un réseau de capteurs sans fil. Nous réduisons ces problèmes en abordant en premier cas une topologie linéaire et ensuite nous généralisons les solutions dans une topologie non linéaire.

5.1.3 Association dans les réseaux de capteurs sans fil

La procédure d'association est un mécanisme utilisé par les nœuds pour rejoindre le réseau. Dans cette partie, nous détaillons les problèmes d'association dans les réseaux de capteurs sans fil utilisant le standard IEEE 802.15.4 [IEE06], et nous proposons deux mécanismes pour les résoudre.

Problèmes d'association

La procédure d'association dans le standard IEEE 802.15.4 change selon le mode utilisé, c'est-à-dire selon que le mode soit avec suivi de balise et sans suivi de balise. Dans le mode avec suivi de balise, un nœud effectue initialement un *scan* (exploration de son voisinage radio) passif, qui consiste à écouter les balises des coordinateurs voisins. Quand ce nœud trouve un coordinateur convenable, il lui envoie une requête d'association. Le coordinateur ainsi contacté répond par un message d'association. Dans le mode sans suivi de balise, un nœud effectue un *scan* actif qui consiste à envoyer une demande de balise aux coordinateurs voisins. Lorsqu'une balise est envoyée par un coordinateur et reçue par un nœud, ce dernier procède comme dans le mode avec suivi de balise et sélectionne un coordinateur convenable. À noter que dans le mode avec suivi de balise, un nœud peut aussi effectuer un *scan* actif s'il n'a pas reçu de balise.

La procédure d'association peut ne pas aboutir pour deux raisons principales : quand un nœud ne détecte aucun coordinateur actif à sa portée, et quand un nœud n'arrive pas à accéder au médium. Ces deux raisons sont largement liées à la charge du médium. En effet, une forte charge du canal diminue la probabilité de succès d'accès au médium et augmente la probabilité de pertes de trames (ce qui contribue à réduire la probabilité de détecter un coordinateur à portée).

Dans [ACC09], les auteurs étudient le pourcentage de nœuds connectés au réseau en présence de plusieurs coordinateurs du PAN mobiles. Ils prouvent que quand les coordinateurs du PAN se déplacent simultanément, le pourcentage de nœuds capteurs connectés est plus petit que quand les coordinateurs du PAN se déplacent à tour de rôle. Ils montrent aussi que le mode sans suivi de balise est plus efficace en termes de temps d'installation et de consommation d'énergie que le mode avec suivi de balise. Cependant, les auteurs n'ont pas évalué le temps d'installation requis pour connecter tous les nœuds du réseau.

Le standard IEEE 802.15.4 ne spécifie pas l'instant d'activation des nœuds lors du déploiement du réseau. Une méthode pour activer les nœuds est de les activer par ordre selon la topologie du réseau. Cette activation est manuelle ce qui n'est pas efficace dans le déploiement de grands

réseaux. Une autre méthode est d'activer les nœuds d'une manière automatique mais arbitraire. Avec cette méthode, des collisions risquent d'apparaître dans le réseau ce qui augmente la durée de déploiement.

Dans ce travail, nous considérons, en simulation, que le standard IEEE 802.15.4 active tous les nœuds presque en même temps. Nous supposons que l'activation de tous les nœuds se fait dans un intervalle de temps court ($[0; 1]$ seconde) indépendant de la topologie. Dans ce qui suit, nous nommons cette activation par l'activation simultanée de nœuds.

Dans cette partie, nous étudions le temps d'installation du réseau en mode avec suivi de balises et nous proposons deux mécanismes afin de réduire significativement ce temps. Pour ce faire, nous étudions tout d'abord le cas particulier des réseaux de capteurs linéaires, puis nous généralisons notre analyse au cas des réseaux de capteurs non linéaires.

Réseaux linéaires. Dans cette partie, nous commençons notre étude sur une topologie spécifique, où les capteurs sont disposés d'une manière linéaire (cf. figure 5.1). Tout d'abord, nous motivons ce choix en donnant quelques exemples de cas où ces topologies apparaissent. Ensuite, nous étudions le comportement de la procédure d'association dans un réseau de capteurs linéaire.

La figure 5.1 montre un exemple de topologie simplifiée d'un réseau linéaire. Le nœud de gauche sur la figure, représenté par un double cercle, est le coordinateur du PAN. C'est lui qui initie l'établissement du réseau. Les grands cercles pointillés représentent la portée des nœuds A et J. Les nœuds sont uniformément distribués et chaque nœud possède seulement deux voisins, à l'exception des nœuds aux extrémités du réseau. C'est cette variante des réseaux linéaires que nous retenons dans ce travail.

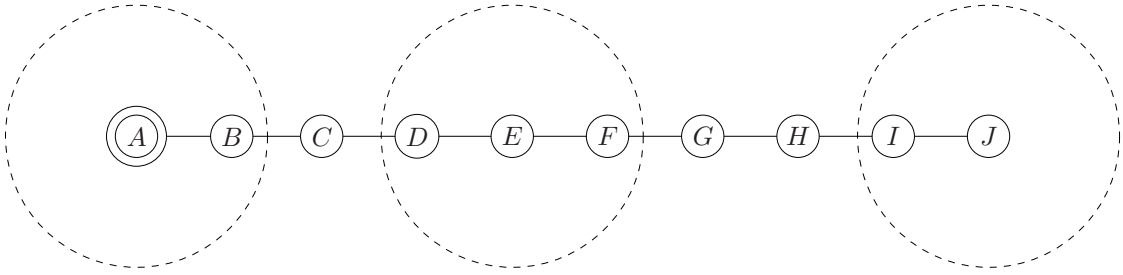


Figure 5.1 – Représentation simplifiée de la topologie d'un réseau linéaire.

Motivation. Dans un réseau linéaire, tous les nœuds sont disposés selon une ligne. Une telle structure linéaire existe dans plusieurs déploiements de capteurs. Par exemple, de longs pipelines [SM07] sont utilisés dans plusieurs pays pour transférer l'eau des usines de dessalement,

qui sont généralement situées à proximité de la mer, aux villes qui sont loin de la mer. Ces pipelines (ou des pipelines de pétrole et de gaz), peuvent être surveillés par des capteurs afin de détecter une trop grande pression ou pour localiser une fuite. Un autre exemple est la surveillance de routes, de ponts et de voies de chemins de fer. Des capteurs peuvent être utilisés pour surveiller les vibrations provoquées par le passage de véhicules, pour assurer une détection précoce de fissures dangereuses le long des structures [LHT+99], pour surveiller la vitesse des véhicules, ou détecter les embouteillages et les accidents. Une discussion détaillée sur les avantages des réseaux linéaires se trouve dans [JMS07].

Activation simultanée des nœuds. Si tous les nœuds sont activés en même temps quand le réseau est déployé, dans ce cas, durant la phase d'installation du réseau, tous les nœuds sont actifs et effectuent des *scans* actifs ou passifs afin de pouvoir se connecter au réseau. Le coordinateur du PAN commence à envoyer des balises pour informer les nœuds à sa portée de son existence. Les nœuds à portée essaient tous de s'associer au réseau en même temps, ce qui génère beaucoup de trafic, surtout si les nœuds effectuent des *scans* actifs. Ce phénomène aboutit à des collisions, et le réseau est congestionné, ce qui augmente la durée de la phase d'installation du réseau. Des résultats sur l'activation simultanée des nœuds sont justifiés dans la partie 5.1.6.

Réseaux non linéaires. À présent que nous avons étudié le cas de topologies simples, nous allons nous intéresser au cas de topologies plus générales, et notamment non linéaires.

La figure 5.2 présente un exemple de topologie non linéaire. Le nœud représenté par un double cercle est le coordinateur du PAN. Il initie l'établissement du réseau. Les grands cercles pointillés représentent la portée de certains nœuds. Les nœuds sont distribués aléatoirement dans le réseau. Par conséquent, le nombre de voisins change d'un nœud à un autre.

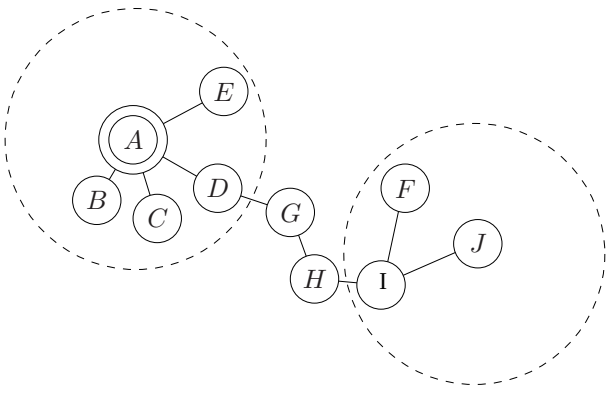


Figure 5.2 – Représentation d'un exemple de topologie non linéaire.

Motivation. Dans les réseaux non linéaires, le déploiement des nœuds dépend énormément de l'application. Dans certaines applications, les nœuds sont aléatoirement distribués. C'est le cas pour la surveillance de sites naturels, où le nombre de nœuds capteurs est très grand. Dans d'autres applications, un déploiement peut être précis, comme par exemple dans le cas du suivi d'une activité industrielle, où le nombre de nœuds est généralement plus petit.

Activation simultanée des nœuds. De même que pour les réseaux linéaires, nous supposons que l'installation des réseaux non linéaires se fait généralement de manière simultanée pour tous les nœuds. Les collisions de messages sont fréquentes, et les zones de congestion peuvent apparaître à plusieurs endroits dans le réseau. Par conséquent, le temps d'installation du réseau augmente. Ceci est justifié dans la partie 5.1.6

5.1.4 Mécanisme SNAIL

Le mécanisme SNAIL *(Sequential Node Activation In a Linear Network)* vise à réduire le temps d'installation du réseau. Bien qu'il puisse être adapté pour les réseaux non linéaires, SNAIL est conçu pour les réseaux linéaires.

Description du mécanisme

Le mécanisme SNAIL consiste à activer les nœuds d'une manière séquentielle. Le but de ce mécanisme est de minimiser le temps d'installation du réseau et maximiser le pourcentage de nœuds associés en un temps donné.

Les principes utilisés par SNAIL sont les suivants. SNAIL suppose que la grande durée d'installation est due au fait que tous les nœuds sont activés simultanément. Désynchroniser les nœuds en les activant un par un peut permettre de réduire la congestion qui est causée par les collisions des paquets de contrôle (balises, message d'associations, etc.), et donc de diminuer le temps d'installation du réseau. De plus, pour éviter qu'un nœud tente de s'associer alors qu'aucun nœud à sa portée n'est activé, ce nœud ne doit pas être activé avant que son prédécesseur ne soit associé au réseau.

SNAIL pour les réseaux linéaires

Supposons que les nœuds sont numérotés de 1 à N en fonction de leur position dans le réseau linéaire. Le premier nœud est le coordinateur du PAN. Selon SNAIL, le coordinateur du PAN est activé à l'instant t_0, et tout nœud n tel que $n \geq 1$ est activé à l'instant $t_n = t_0 + p + \alpha(n-1)$,

où p représente le temps d'activation du coordinateur du PAN et $\alpha \geq p$ désigne le temps moyen dont un nœud a besoin pour être associé à son prédécesseur.

La figure 5.3 montre la procédure d'activation des nœuds selon le mécanisme SNAIL. La flèche partant du coordinateur du PAN A et allant jusqu'au dernier nœud du réseau J indique que les nœuds sont activés séquentiellement.

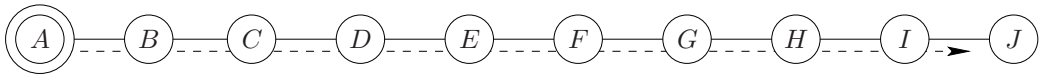


Figure 5.3 – Dans une topologie linéaire, SNAIL active les nœuds séquentiellement.

Nous avons estimé par simulation la valeur de p, qui est approximativement égale à 1 seconde. Nous avons l'intention de trouver la valeur de α de la même manière. La valeur de ce paramètre ne doit pas dépendre de la taille du réseau, puisque la charge du trafic pour la phase d'installation d'un réseau linéaire est uniformément répartie. En effet, durant la phase d'installation du réseau, la zone de congestion se déplace du coordinateur du PAN jusqu'au dernier nœud associé [JS03] en mode avec suivi de balise. Cependant, dans le mode avec suivi de balise, la valeur de α dépend de la durée de l'intervalle entre deux balises, qui est fixée par le paramètre BO. Quand le réseau fonctionne en mode sans suivi de balise, le paramètre α peut être assimilé à une constante.

Implémentation dans les réseaux linéaires

Dans la littérature, plusieurs mécanismes de localisation existent et peuvent être utilisés dans les réseaux de capteurs sans fil afin qu'un nœud puisse estimer sa position dans le réseau [MFA07]. Cependant, nous ne pouvons pas utiliser ces mécanismes dans notre cas puisque nous concentrons notre étude sur la période avant que le réseau ne soit complètement opérationnel (la phase d'installation est une phase durant laquelle aucun paquet de données n'est transmis). De plus, nous considérons qu'il n'est pas possible que les nœuds connaissent leur position (que ce soit en ayant ces valeurs configurées au moment du déploiement, ou avec un système de positionnement global).

Le nœud n_i, qui est le $i^{\text{ème}}$ nœud du réseau, doit calculer son temps d'association $t_i = t_0 + p + (i-1)\alpha$, où p et α sont des constantes connues *a priori*.

Puisque le réseau est linéaire, un nœud peut démarrer la procédure d'association si et seulement si son prédécesseur n_{i-1} est associé au réseau. Le temps t_{i-1} peut être déterminé en entendant les trames d'associations de n_{i-1}. Le nœud n_i n'a pas le droit de transmettre de requêtes de balise ni de requêtes d'association tant qu'il n'a pas entendu des trames d'associa-

tions de n_{i-1}. Quand n_i est informé de l'association de n_{i-1} au réseau, il attend pour α secondes avant de commencer sa propre procédure d'association.

SNAIL pour les réseaux non linéaires

Dans les réseaux non linéaires, le mécanisme SNAIL trie les nœuds selon leur profondeur dans le réseau. La profondeur d'un nœud correspond au nombre de sauts pour arriver au coordinateur du PAN. Tous les nœuds ayant une profondeur d sont activés avant que le premier nœud de profondeur $d+1$ ne soit activé. En outre, les nœuds possédant la même profondeur sont activés séquentiellement. L'intervalle de temps entre l'activation de deux nœuds est la constante α.

La figure 5.4 montre un exemple du mécanisme SNAIL dans un réseau non linéaire. Pour une simplicité de présentation, nous avons considéré un exemple de 25 nœuds uniformément répartis pour couvrir une surface de 50 m × 50 m avec une portée de 10 m et donc chaque nœud possède 4 voisins au maximum. Le coordinateur du PAN, représenté par un double cercle, est situé au centre du réseau. La flèche partant du coordinateur du PAN représente l'ordre d'activation des nœuds selon le mécanisme SNAIL. La procédure d'activation des nœuds commence par le coordinateur du PAN et ensuite tourne en spirale en s'éloignant du coordinateur du PAN.

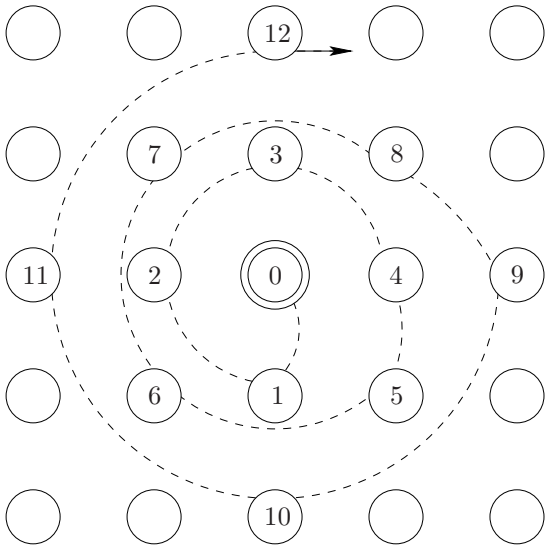


Figure 5.4 – Dans une topologie non linéaire, SNAIL active les nœuds séquentiellement selon un ordre défini pour la topologie.

Utiliser SNAIL dans des réseaux non linéaires permet de réduire les collisions dans le réseau, plus précisément dans la phase d'installation du réseau. Par contre, le temps d'installation du réseau est grand, surtout lorsque le réseau est grand car les nœuds sont activés séquentiellement.

5.1.5 Mécanisme *Bull's Eye*

Dans cette partie, nous proposons le mécanisme d'association *Bull's Eye*, qui est adapté pour les réseaux non linéaires. Ce mécanisme peut être vu comme une amélioration du mécanisme SNAIL.

Description du mécanisme

Bull's Eye consiste à activer les nœuds d'une manière quasi-séquentielle, ce qui permet de maximiser le nombre de nœuds qui peuvent s'associer au réseau et de minimiser le temps de la phase d'installation du réseau.

Avec le mécanisme *Bull's Eye*, tous les nœuds ayant la même profondeur sont activés simultanément, et chaque profondeur est activée séquentiellement (en partant de la profondeur 0 du réseau qui correspond au coordinateur du PAN). Le mécanisme *Bull's Eye* est une approche hybride combinant à la fois l'approche qui consiste à activer les nœuds simultanément, et SNAIL qui consiste à activer les nœuds séquentiellement.

Bull's Eye pour les réseaux non linéaires

Le temps d'activation pour un nœud n ayant une profondeur d est donnée par $t_n = t_0 + p + \beta d$, où t_0 est le temps d'activation du coordinateur du PAN, $p = 1$ seconde (d'après l'étude précédente) et β est l'intervalle de temps séparant deux activations de profondeurs successives. Nous proposons que β soit constant afin de bénéficier du phénomène de parallélisme lors de l'activation des nœuds ayant la même profondeur, même si le nombre de nœuds à chaque profondeur d n'est pas le même.

La figure 5.5 montre l'ordre d'activation des nœuds selon le mécanisme *Bull's Eye*, pour un réseau de 25 nœuds. Pour faciliter la présentation du mécanisme, nous utilisons un réseau où les nœuds sont répartis en grille pour couvrir une surface de 50 m × 50 m. Le coordinateur du PAN est situé au centre du réseau. Comme nous pouvons le voir sur la figure, les premiers nœuds à être activés après le coordinateur du PAN sont les nœuds qui lui sont le plus proches. Ensuite, les nœuds sont activés selon des cercles (représentés en pointillés) de diamètres croissants. Le nombre de nœuds à profondeur d augmente généralement avec d, ce qui permet de profiter de l'éloignement géographique des nœuds pour le parallélisme. En effet, si tous les nœuds à 1 saut sont à portée les uns des autres, les transmissions parallèle aboutissent à des collisions. Par contre, les nœuds à plus d'un saut ne sont pas probablement tous à portée les uns des autres et il est donc possible pour certains nœuds de transmettre simultanément des balises et des

requêtes d'association sans produire beacoup d'interférence. Le temps nécessaire pour que tous les nœuds transmettent n'est donc pas linéaire avec le nombre des nœuds.

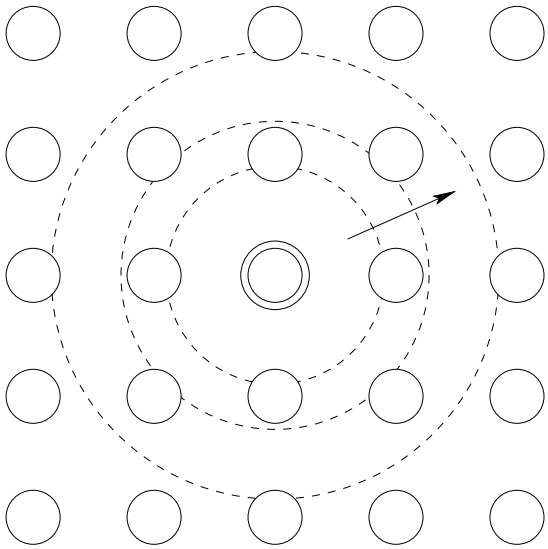


Figure 5.5 – Dans une topologie non linéaire, *Bull's Eye* associe les nœuds de chaque profondeur parallèlement.

Implémentation dans les réseaux non linéaires

Un nœud n à une profondeur d (en terme de nombre de sauts) du coordinateur du PAN doit calculer son temps d'association $t_n = t_0 + p + d\beta$, où p et β sont des constantes connues *a priori*.

Une piste à étudier serait la suivante. Un nœud peut écouter les échanges de trames avant t_n, bien qu'il n'ait pas le droit de commencer la procédure d'association avant t_n. Quand le nœud n reçoit une trame f d'un autre nœud s, n peut extraire quelques données de la trame f (même si la destination de cette trame n'est pas n). Si f fait partie de la procédure d'association, n peut déterminer la profondeur de s [1] et donc la sienne.

5.1.6 Résultats

Cette partie met en relief les résultats de simulations obtenus pour comparer le mécanisme d'activation simultanée des nœuds avec les deux nouveaux mécanismes que nous avons proposés. Tout d'abord, nous discutons des résultats obtenus sur les réseaux linéaires, puis de ceux des réseaux non linéaires. Ces résultats sont reliés aux métriques indispensables pour la phase

1. La détermination de l'adresse du nœud s peut être faite en utilisant les propriétés de l'allocation d'adresses hiérarchique utilisé dans ZigBee, ou bien en ajoutant un champ pour la profondeur du nœud émetteur dans la trame d'association du standard IEEE 802.15.4.

d'installation des réseaux qui sont le temps d'installation du réseau et le nombre de nœuds qui peuvent s'associer à ce réseau. Nous faisons varier le nombre de nœuds dans le réseau de 9 à 81 nœuds sur des topologies linéaires ou en grille. La distance entre nœuds adjacents est fixée à 10 m. Les simulations sont lancées 100 fois, et les résultats présentés sont la moyenne de ces répétitions.

Nous considérons qu'une association ne s'achève pas si certains nœuds n'arrivent pas à s'associer au réseau dans un délai de 2000 secondes. Les simulations sont réalisées sous NS-2 [NS202].

Réseaux linéaires

Nous avons fixé la portée des nœuds dans les réseaux linéaires de sorte que chaque nœud possède deux voisins (un à sa gauche et un à sa droite). Le coordinateur du PAN est toujours situé à un bout du réseau.

Temps d'installation d'un réseau linéaire. La figure 5.6 montre les résultats de simulations obtenus pour le temps d'installation du réseau, en fonction de la taille du réseau et du paramètre BO. Pour des valeurs de BO supérieures à 8, les nœuds n'arrivent pas à s'associer en moins de 2000 secondes. Rappelons que la durée du cycle correspondant à BO = 8 est de $3,932$ secondes. Comme prévu, le temps nécessaire pour achever la phase d'installation d'un réseau augmente avec le nombre de nœuds. Cependant, la valeur de BO affecte négativement le temps d'installation. Puisque l'intervalle de temps séparant la transmission de deux balises est une fonction exponentielle de BO, et puisque l'association nécessite que les nœuds reçoivent des balises, le temps d'installation augmente lui aussi exponentiellement avec BO. Il est important de noter sur cette figure que le temps n'est pas une fonction linéaire de la taille du réseau. En effet, puisque tous les nœuds essaient de s'associer au réseau en même temps, ils génèrent beaucoup de trafic, et particulièrement quand ils décident d'effectuer un *scan* actif.

Nombre de nœuds associés au réseau pour un temps d'installation fixe. La figure 5.7 montre le pourcentage de nœuds qui sont associés au bout d'une durée de 2000 secondes. Ce pourcentage est calculé en fonction de BO pour un réseau de 81 nœuds. Par exemple, quand BO est égal à 7, 15% des nœuds seulement sont associés en utilisant l'approche d'association de base. SNAIL avec $\alpha = 1$ réussit à associer presque tous les nœuds pour toute valeur de BO entre 4 et 8. Quand BO est égal à 3, $\alpha = 2$ assure un très haut pourcentage de nœuds associés.

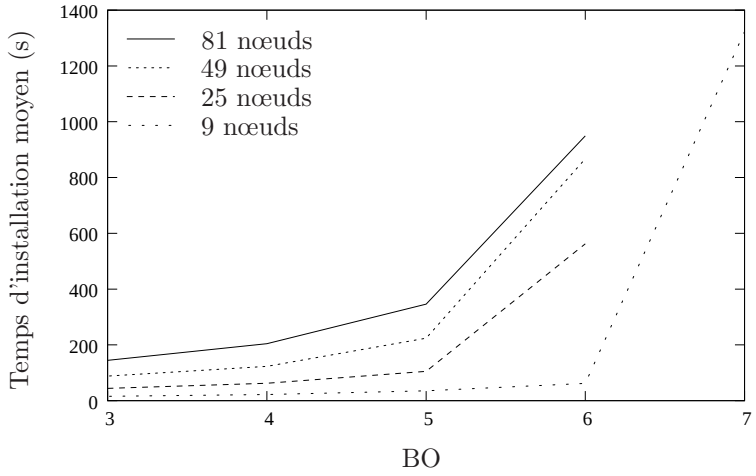


Figure 5.6 – Le temps d'installation pour un réseau linéaire dépend du nombre de nœuds du réseau et de BO.

Puisque le pourcentage de nœuds associés à un réseau durant 2000 secondes augmente, le temps d'installation du réseau diminue. SNAIL répond donc au problème de la phase de déploiement du réseau et diminue son temps d'installation.

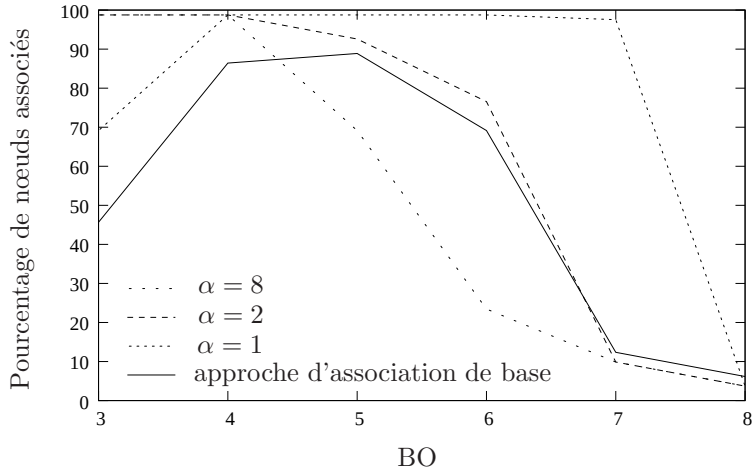


Figure 5.7 – SNAIL augmente significativement le pourcentage de nœuds associés.

Réseaux non linéaires

À présent, nous considérons des structures de réseaux plus générales. Pour faciliter l'analyse des résultats, nous étudions des réseaux en grille, où les nœuds sont uniformément distribués. Cependant, les résultats peuvent être très proches pour des réseaux aléatoirement distribués.

Temps d'installation d'un réseau non linéaire. La figure 5.8 montre le temps d'installation nécessaire pour associer tous les nœuds à un réseau avec l'approche d'association de

base. Par exemple, si le réseau est composé de 49 nœuds, la durée de la phase d'installation est approximativement 300 secondes quand BO est égal à 6. Quand les nœuds sont activés selon l'approche d'association de base, les nœuds associés envoient des balises alors que les nœuds non associés au réseau effectuent des *scans* actifs et envoient des requêtes de balises ou des requêtes d'association. Des collisions apparaissent et deviennent plus significatives pour des réseaux de grande taille. Les résultats de simulation montrent que, pour un réseau de 81 nœuds et pour une valeur de BO égale ou supérieure à 6, certains nœuds ne sont jamais associés au réseau.

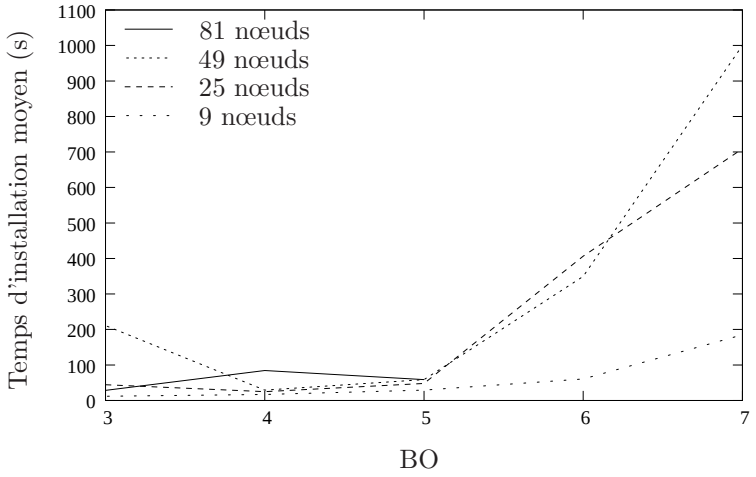


Figure 5.8 – Le temps d'installation est élevé quand tous les nœuds sont activés selon l'approche d'association de base (c'est-à-dire simultanément).

SNAIL dans un réseau non linéaire. La figure 5.9 illustre le temps d'installation pour SNAIL dans un réseau de 81 nœuds, en fonction du paramètre BO. Le paramètre α a un impact sur les performances du mécanisme SNAIL, et il n'existe aucune valeur du paramètre α qui permet à SNAIL d'améliorer les performances de l'approche d'association de base. Ceci est dû au fait que SNAIL ne bénéficie pas de l'activation parallèle des nœuds. Deux nœuds distants ayant la même profondeur (par exemple, des nœuds situés de part de d'autre de la topologie) ne peuvent pas être activés en même temps, alors qu'ils pourraient s'associer au réseau sans causer de collisions.

Bull's Eye **dans un réseau non linéaire.** Le mécanisme *Bull's Eye* est proposé pour améliorer le mécanisme SNAIL pour les réseaux non linéaires. Pour cela, nous avons lancé des simulations avec les mêmes paramètres que ceux utilisés dans les parties précédentes. La figure 5.10 montre le temps d'installation avec *Bull's Eye* dans un réseau de 81 nœuds, en fonction du paramètre BO. Quand le paramètre BO est inférieur ou égal à 5, toutes les valeurs du paramètre

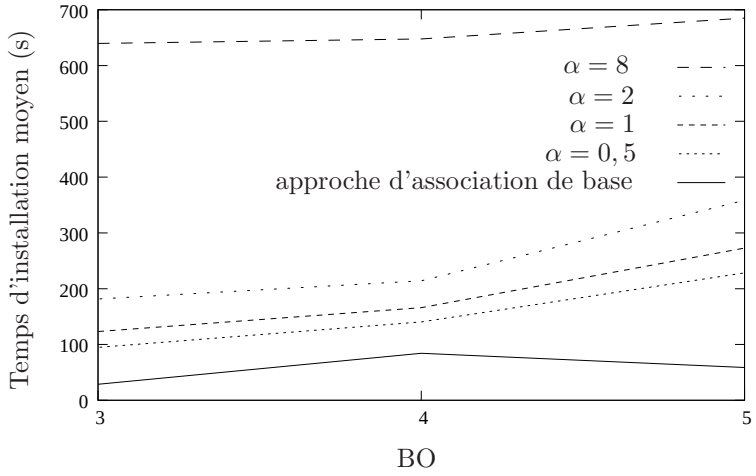


Figure 5.9 – SNAIL n'arrive pas à améliorer le temps d'installation d'un réseau non linéaire par rapport à l'approche d'association de base.

β réduisent le temps d'installation par rapport à l'approche d'association de base. Par exemple, quand BO est égal à 4, *Bull's Eye* réduit de 37% (pour $\beta = 8$) à 85% (pour $\beta = 0,5$) le temps d'installation de l'approche d'association de base. Quand BO est strictement supérieure à 5, l'approche d'installation de base du réseau est incapable d'associer les 81 nœuds au réseau. Ceci est dû aux collisions des balises. Cependant, *Bull's Eye* est capable d'associer tous les nœuds, avec un délai raisonnable.

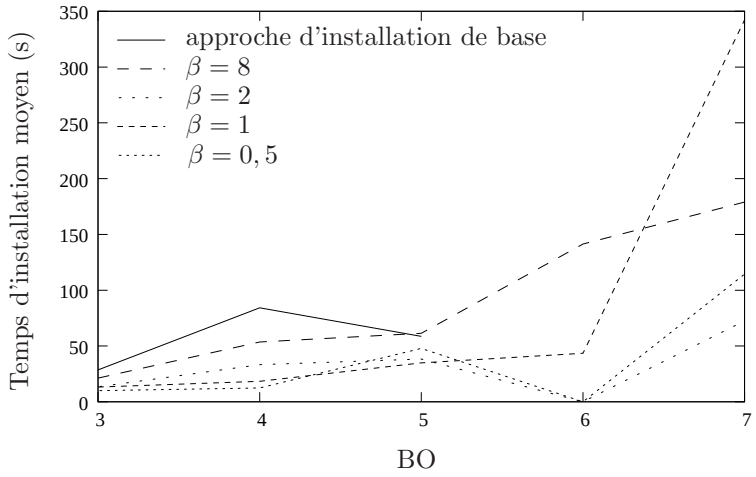


Figure 5.10 – Le mécanisme *Bull's Eye* améliore l'approche d'installation de base.

5.1.7 Allocation d'adresses dans les réseaux de capteurs sans fil

La procédure d'allocation d'adresses des nœuds est un mécanisme indispensable au réseau, qui est effectué pendant l'association. L'allocation d'adresses consiste à donner des adresses aux nœuds associés au réseau afin de pouvoir les identifier durant la transmission et la réception

des données. Dans cette partie, nous détaillons les différents problèmes d'allocation d'adresses qui peuvent avoir lieu dans un réseau et nous proposons un mécanisme d'allocation d'adresses capable de les résoudre.

Problèmes d'allocation d'adresses

Le mécanisme d'allocation d'adresses se fait au niveau de la couche réseau de la pile protocolaire d'un réseau de capteurs sans fil. Avant son association au réseau, un nœud est identifié par son adresse MAC, appelée adresse longue. Cette adresse occupe 8 octets. La couche réseau alloue à chaque nouveau nœud une adresse logique, plus courte que l'adresse MAC car occupant 2 octets seulement. Ceci permet à la sous-couche MAC (notamment) de réduire la taille du champ d'adresses dans les trames échangées.

Comme nous l'avons déjà vu dans le chapitre 1, il existe deux mécanismes d'allocation d'adresse pour les réseaux de capteurs sans fils utilisant le standard ZigBee : le mécanisme d'allocation d'adresses distribué et le mécanisme d'allocation d'adresses aléatoire. Ces deux mécanismes sont limités par des problèmes que nous détaillons dans la suite de cette partie. Nous explorons tout d'abord les problèmes du mécanisme d'allocation d'adresses distribué DAAM *(Distributed Address Allocation Mechanism)*. Ensuite, nous identifions les problèmes du mécanisme d'allocation d'adresses aléatoire SAAM *(Stochastic Address Allocation Mechanism)*.

Problèmes du mécanisme d'allocation d'adresses DAAM. Dans le mécanisme d'allocation d'adresses DAAM, les adresses sont attribuées selon l'arbre construit par ZigBee. Dans cet arbre, chaque père possède au plus C_m fils dont R_m coordinateurs fils, et la hauteur maximale de l'arbre est L_m.

Problème 1. *Si les paramètres du réseau ne sont pas bien choisis, il est possible qu'un nœud n'arrive pas à s'associer, alors qu'il existe des adresses disponibles.*

Le problème 1 a lieu quand un nœud essaie de rejoindre le réseau, mais est à portée de nœuds FFD qui ont déjà C_m fils.

Problème 2. *Si le réseau n'est pas uniformément déployé, plusieurs adresses peuvent ne pas être attribuées.*

Le problème 2 est dû au fait que dans les réseaux qui ne sont pas uniformément déployés, les paramètres doivent être grands pour pouvoir tenir compte des zones de grandes densités. Dans

les zones de faibles densités, les adresses sont pré-réservées, mais elle ne sont pas attribuées aux nœuds (car ils sont trop peu nombreux).

La profondeur maximale d'un arbre L_m est un autre paramètre critique, puisqu'il est utilisé dans la formule de $Cskip$ (cf. partie 1.3.2) comme un exposant. Connaissant C_m et R_m, nous pouvons déterminer la valeur maximale de la profondeur de l'arbre L_m^{max} selon la formule suivante :

$$L_m^{max} = \max_{L_m}\{\#\text{dev}(C_m, R_m, L_m) \leq 2^{16}\},$$

où #dev est une fonction qui retourne le nombre total de nœuds qui peuvent être associés à un réseau pour C_m, R_m et L_m. $\#\text{dev}(C_m, R_m, L_m)$ est égal à la somme du nombre maximal de nœuds pour chaque profondeur $d \leq L_m$, ce qui veut dire :

$$\#\text{dev}(C_m, R_m, L_m) = 1 + \sum_{1 \leq d \leq L_m - 1} (R_m^d + R_m^{d-1}(C_m - R_m)).$$

Le tableau 5.1 montre les valeurs de L_m^{max} pour différentes valeurs de C_m et R_m. Nous remarquons que L_m^{max} est toujours inférieur ou égal à 17, même pour de petites valeurs de C_m et R_m. Dans un environnement de mines, par exemple, le réseau est déployé sur des centaines de mètres et la portée des communications est limitée à quelques mètres : la profondeur du réseau doit être élevée. Une telle limitation de profondeur d'arbre à 17 interdit l'utilisation du mécanisme DAAM.

C_m	R_m	L_m^{max}	C_m	R_m	L_m^{max}
2	2	17	3	2	16
4	2	16	5	2	15
3	3	11	4	3	11
4	4	9	5	4	9

Tableau 5.1 – L_m^{max} est toujours inférieur ou égal à 17, même pour de petites valeurs de C_m et R_m.

Problèmes du mécanisme d'allocation d'adresses SAAM. Avec le mécanisme SAAM, il est possible que deux nœuds aient la même adresse puisque les adresses sont aléatoirement attribuées. Ce problème est relié au principe des casiers *(pigeonhole)* [Gri98], principe paradoxe et au problème des anniversaires [McK66].

Le principe des casiers affirme que si n pigeons choisissent un casier parmi m, avec $n > m$, alors il existe au moins un casier avec $\lceil n/m \rceil$ pigeons.

Le paradoxe des anniversaires est à l'origine une estimation probabiliste du nombre de personnes n qu'il faut regrouper pour avoir une probabilité supérieure ou égale à $0,5$ que deux personnes aient le même jour d'anniversaire parmi les m jours de l'année (avec $m = 365$ jours). Ce paradoxe est connu car la probabilité dépasse $0,5$ quand $n = 23$.

Nous allons calculer le nombre moyen de conflits d'adresses quand n nœuds choisissent aléatoirement une adresse parmi $a = 2^{16}$ adresses, en nous basant sur le principe des casiers et sur le paradoxe des anniversaires.

Propriété 4. *Le nombre d'adresses différentes, quand n nœuds choisissent leur adresse aléatoirement parmi a et que k adresses différentes ont été précédemment allouées, est :*

$$D(n,k,a) = kq^n + \frac{1-q^n}{1-q},$$

avec $q = 1 - 1/a$.

Démonstration. Considérons tout d'abord le cas où $n = 0$. Le nombre d'adresses différentes et uniques est égal à k, selon la définition de k. Maintenant, définissons $D(n+1,k,a)$ en fonction de $D(n,k,a)$. Quand le $(n+1)$ème nœud choisit une adresse aléatoire, il peut choisir une adresse qui est déjà prise par un autre nœud, ou bien une adresse qui n'est pas encore utilisée. La probabilité de choisir une adresse qui est déjà attribuée pour un autre nœud est égale à $D(n,k,a)/a$. Dans ce cas, $D(n+1,k,a) = D(n,k,a)$, puisque le nombre d'adresses différentes reste constant. La probabilité de choisir une adresse qui n'est pas attribuée à un autre nœud est égale à $1 - D(n,k,a)/a$. Dans ce cas, $D(n+1,k,a) = D(n,k,a) + 1$, puisque la nouvelle adresse est différente des adresses précédentes. En définissant $q = 1 - 1/a$, nous avons :

$$\begin{aligned} D(n+1,k,a) &= D(n,k,a) \cdot D(n,k,a)/a + (D(n,k,a)+1) \cdot (1 - D(n,k,a)/a) \\ &= D(n,k,a)^2/a - D(n,k,a)^2/a + 1 + D(n,k,a) - D(n,k,a)/a \\ &= D(n,k,a)q + 1. \end{aligned}$$

Algorithme 4 Nombre moyen de conflits d'adresses quand SAAM attribue des adresses aléatoires pour n nœuds.

Paramètres : n est le nombre de nœuds
Retourne : le nombre moyen de conflits

1: $a \leftarrow 2^{16}$
2: $k \leftarrow 0$
3: $n_{\text{initial}} \leftarrow n$
4: $conflits \leftarrow 0$
5: $n_{\text{alloués}} \leftarrow 0$
6: **tantque** $n_{\text{alloués}} \neq n_{\text{initial}}$ **faire**
7: $\quad n_{\text{alloués}} \leftarrow D(n, k, a)$
8: $\quad conflits \leftarrow conflits + n_{\text{initial}} - n_{\text{alloués}}$
9: $\quad k \leftarrow n_{\text{alloués}}$
10: $\quad n \leftarrow n_{\text{initial}} - n_{\text{alloués}}$
11: **fin tantque**
12: **Retourner** $conflits$

Cette expression récursive de $D(n, k, a)$ peut être simplifié de la manière suivante :

$$\begin{aligned} D(n,k,a) &= D(n-1,k,a)q + 1 \\ &= D(n-2,k,a)q^2 + q + 1 \\ &= \dots \\ &= D(0,k,a)q^n + \sum_{i=0}^{n-1} q^i \\ &= kq^n + \frac{1-q^n}{1-q}. \end{aligned}$$

□

Le nombre moyen de conflits peut être obtenu à partir de $D(n, k, a)$ en raisonnant par récurrence. À la première étape, chacun des n nœuds doit choisir une adresse parmi a, avec $k = 0$ adresses uniques pré-allouées. Ceci aboutit à $D(n, k, a)$ adresses différentes à la fin de la première étape. À la deuxième étape, chacun des $n - D(n, k, a)$ nœuds en conflit doit choisir une adresse parmi a sachant que $D(n, k, a)$ adresses uniques sont déjà pré-réservées. Ce processus continue jusqu'à ce qu'il n'y ait plus de conflit dans le réseau. Le calcul du nombre moyen de conflits attendu est donné dans l'algorithme 4.

La figure 5.11 montre l'évolution du nombre moyen de conflits en fonction du nombre total de nœuds dans le réseau. Nous supposons que les nœuds utilisent un générateur aléatoire parfait [2]. Même si le nombre de nœuds dans le réseau est relativement petit, le nombre de conflits n'est

2. Cette hypothèse n'est pas vraie en pratique, et le nombre moyen de conflits dans un cas réel est supérieur aux valeurs illustrées dans la figure 5.11.

pas négligeable.

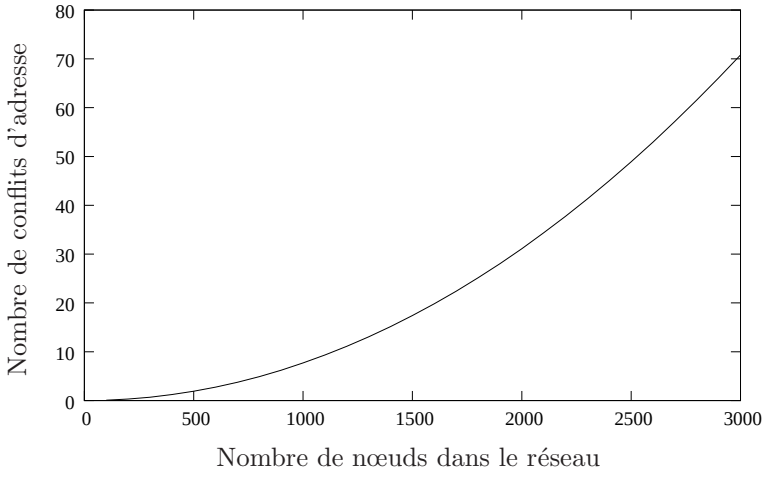


Figure 5.11 – Le nombre de conflits pour n nœuds augmente d'une manière parabolique.

Notons que les conflits nécessitent du temps afin d'être détectés et résolus. Quand un conflit est détecté pour un nœud, toutes les communications impliquant ce nœud sont interrompues, ce qui ne peut pas être toléré dans un réseau déployé dans une mine où la sécurité des ouvriers est une contrainte importante.

Mécanisme d'allocation d'adresses binaire BAAM

Nous proposons à présent un mécanisme d'allocation d'adresses pour attribuer des adresses aux nœuds du réseau. Ce mécanisme est nommé BAAM *(Binary Address Assignment Mechanism)*. BAAM gère des réseaux de grande profondeur tout en maintenant l'avantage principal de DAAM qui consiste à fournir une fonctionnalité de routage. BAAM ne possède pas les inconvénients de SAAM puisqu'il utilise un mécanisme déterministe.

Description du mécanisme. BAAM suppose qu'un nœud FFD peut avoir seulement deux fils FFD, et que la topologie est majoritairement linéaire. BAAM se base sur les propriétés suivantes :

1. deux fils d'un même nœud peuvent se voir attribuer des plages d'adresses de longueur différentes,
2. le calcul du prochain saut est fait en local, et non pas avec une formule globale (comme le $Cskip$),
3. les adresses du second fils sont attribuées à partir de la fin de la plage d'adresses.

Ces propriétés permettent à BAAM de réutiliser la plupart des adresses non utilisées.

Allocation d'adresses de BAAM. Dans BAAM, la plage d'adresses pour chaque FFD varie. Initialement, le coordinateur du PAN se voit attribuer toute la plage d'adresses. Quand un nœud FFD y de père x attribue une adresse à son fils z, y commence à attribuer les adresses à partir du début de sa plage d'adresses, si y est le premier fils de x, ou bien à partir de la fin de sa plage d'adresses si y est le second fils de x. Quand un nœud FFD détecte qu'il n'a pas deux fils, et qu'il perd potentiellement des adresses de sa plage d'adresses, il peut redistribuer quelques unes des adresses non utilisées à son père ou à son fils (s'il en a un).

La figure 5.12 montre un exemple de BAAM, où le nombre maximal d'adresses est fixée à 21 au lieu de 2^{16}, pour des raisons de simplicité. Le coordinateur du PAN possède l'adresse 0, comme dans le mécanisme DAAM. Ensuite, le coordinateur du PAN divise sa plage d'adresses $[0; 20]$ en deux parties égales : $[1; 10]$ et $[11; 20]$, et donne la première partie à son premier fils FFD. La plage d'adresses non utilisées, $[11; 20]$, peut être utilisée par la suite pour étendre la plage d'adresses du premier fils FFD, à condition qu'aucun second nœud fils ne soit associé à 0. Le nœud ayant l'adresse 1 divise aussi sa plage d'adresses $[1; 10]$ en deux parties égales : $[2; 5]$ et $[6; 10]$, et donne chacune à l'un de ses deux fils. Le premier fils utilise la première adresse de sa plage d'adresses, qui est l'adresse 2, comme sa propre adresse. Le second fils utilise la dernière adresse de sa plage d'adresse, qui est l'adresse 10, comme sa propre adresse. Si un grand nombre de nœuds essaient de s'associer au nœud 2 ou bien au nœud 10, ces deux nœuds peuvent demander à leur père, le nœud 1 d'étendre leur plage d'adresses.

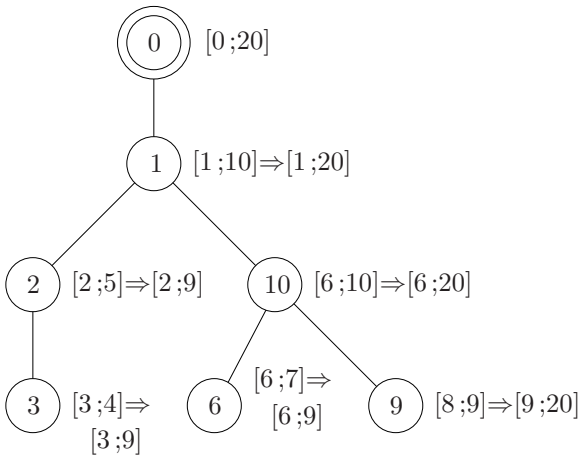


Figure 5.12 – La plage d'adresses actuelle (côté gauche de ⇒) peut être étendue (côté droite de ⇒) par le père ou bien par l'un des fils.

Pour augmenter la plage d'adresses, le nœud 1 peut effectuer l'une des deux actions suivantes.

– Le nœud 1 peut demander à la branche de son fils 2 à redonner les adresses non utilisées. Le nœud 2 peut identifier la plage d'adresses non utilisée, $[4; 5]$, et la rendre à son père,

le nœud 1, ou bien, par un processus récursif, demander au nœud 3 de réduire sa plage d'adresses. Ensuite, le nœud 1 peut changer la plage d'adresses du nœud 2 en $[2; 4]$, et augmenter la plage d'adresses de 10 en $[5; 10]$,

– Le nœud 1 peut demander à son père, le nœud 0, de rendre les adresses non utilisées. À son tour, le nœud 0 peut redistribuer ses plages d'adresses de la manière suivante : le nœud 1 se voit donner la plage d'adresses $[1; 17]$ et le nœud 0 conserve la plage d'adresses $[18; 20]$ par exemple, au cas où un futur nœud FFD souhaite s'associer au réseau.

Avantage de BAAM. BAAM est capable de redonner une partie des adresses qui sont non utilisées par les fils ou par le père d'un nœud n. De cette manière, un nœud peut associer de nouveaux nœuds qui demandent à être associés au réseau, même si sa plage actuelle d'adresses a été totalement attribuée aux nœuds existant dans le réseau. Cette amélioration est faite en s'assurant que le routage hiérarchique est toujours possible.

5.1.8 Synthèse

Dans la partie 5.1, nous avons traité la congestion durant la phase d'installation du réseau. Cette phase réalise deux tâches : l'association et la distribution des adresses aux nœuds.

Nous avons tout d'abord analysé le temps nécessaire pour déployer un réseau avant qu'il ne soit complètement opérationnel. Ensuite, nous avons identifié les problèmes d'association ainsi que les problèmes d'adressage qui augmentent la congestion dans le réseau, et par conséquent augmentent la durée de la phase d'installation. Nous avons proposé deux mécanismes, SNAIL et *Bull's Eye*, pour réaliser des associations rapides. De même, nous avons proposé le mécanisme BAAM, qui réalise l'allocation d'adresses aux nœuds.

5.2 Congestion sur les chemins de communications

Nous traitons à présent le problème de congestion qui apparaît durant la phase de communications. Dans cette partie, nous proposons des solutions qui contribuent à éviter la congestion dans le réseau.

L'efficacité des protocoles de routage dans un réseau de capteurs sans fil est généralement mesurée en termes de faible taux de pertes de paquets et de faible délai de bout-en-bout. Les bonnes performances sont obtenues en réduisant la congestion du médium.

Quand un événement urgent est détecté dans un réseau de capteurs sans fil, les nœuds qui sont situés dans la zone d'urgence transmettent des données urgentes avec un taux de

transmission élevé. Ces données urgentes, que nous appelons alarmes, doivent être traitées d'une façon plus prioritaire que le trafic normal. Ceci peut être fait en utilisant des protocoles de routage efficaces qui réduisent le taux de pertes des paquets et le délai de bout-en-bout.

Les protocoles de routage réactifs, comme AODV, ne sont pas convenables pour le routage des alarmes. En effet, les protocoles réactifs nécessitent d'établir un chemin des sources de la zone d'alarmes vers la destination, avant que les paquets d'alarmes ne puissent être acheminés. Le temps requis pour établir ce chemin est significatif puisqu'il dépend de la distance entre ces sources et la destination, et de la taille du réseau.

Les protocoles de routage proactifs déterministes sont capables de router les alarmes dès qu'elles sont produites, mais causent de la congestion sur la majorité du chemin jusqu'à la destination. En effet, tous les nœuds situés dans la zone d'alarmes envoient des paquets d'alarmes à la destination en suivant des chemins qui convergent rapidement. Les nœuds situés sur la partie commune de ces chemins doivent router les paquets de plusieurs sources. Cet état du réseau cause une large contention pour l'accès au médium, et donc, les zones autour de ces nœuds et les parties communes des chemins sont congestionnées.

La congestion a deux impacts majeurs dans un réseau. Premièrement, des paquets (et notamment des paquets d'alarmes) peuvent être perdus si le trafic est élevé. Deuxièmement, la congestion aboutit à augmenter la consommation d'énergie puisqu'elle cause des collisions de paquets, et par conséquent des retransmissions.

5.2.1 Protocole de routage pour un réseau mono-puits

PiRAT *(Pivot Routing for Alarm Transmission)* [ERGM09b] est un protocole de routage basé sur des nœuds pivots pour la transmission d'un trafic d'alarme. Il peut être adapté pour tout type de trafic à haute priorité généré dans une zone déterminée. PiRAT améliore le protocole de routage raccourci de l'arbre en introduisant une diversité dans le routage et en diminuant les zones de congestion dans le réseau.

Dans cette partie, nous décrivons PiRAT et ses caractéristiques. Ensuite, nous détaillons le mécanisme de sélection des pivots. Nous proposons une formulation linéaire théorique et nous considérons une heuristique pour la sélection des pivots. Par la suite, nous détaillons le protocole de découverte des pivots et nous étudions le comportement de PiRAT pour des réseaux à faibles taux d'activité.

Description du protocole PiRAT

PiRAT vise à réduire la congestion créée par la transmission d'alarmes dans un réseau de capteurs sans fil. Pour réaliser ceci, PiRAT nécessite deux étapes. La première étape consiste à sélectionner des nœuds spéciaux, appelés pivots, pour chaque paire source-destination. Le rôle des pivots est de distribuer la charge du trafic sur plusieurs nœuds au lieu de surcharger le plus court chemin reliant les sources (généralement localisées dans une zone délimitée du réseau) à la destination. La deuxième étape consiste à acheminer les paquets d'alarmes à travers un nœud pivot aléatoirement choisi parmi l'ensemble des pivots. Ainsi, PiRAT est donc un protocole proactif probabiliste.

PiRAT est basé sur la diversité des routes grâce à la nature probabiliste de la sélection des pivots. Il s'appuie sur le protocole de routage raccourci de l'arbre : les routes de la source jusqu'au pivot et du pivot jusqu'à la destination sont calculées selon le protocole de routage raccourci de l'arbre. PiRAT possède une autre caractéristique probabiliste : quand plusieurs voisins conduisent à la destination finale par un chemin de même distance, et que cette distance est minimale, PiRAT choisit aléatoirement un nœud parmi ces voisins afin de router l'alarme.

Le principal avantage de PiRAT est qu'il assure un routage multi-chemins jusqu'à la destination. En effet, il permet à un grand nombre de nœuds de participer à l'activité de routage. Ainsi, la consommation d'énergie est équilibrée entre les nœuds du réseau. Cette caractéristique a l'avantage de prolonger la durée de vie du réseau.

Dans [Bei08], l'auteur propose une approche centralisée où k chemins sont établis de la source à la destination. Chaque chemin passe par plusieurs pivots, mais la distance entre les pivots est limitée par un seuil. Cet algorithme requiert une connaissance globale du réseau. PiRAT est lui un protocole de routage distribué. Dans PiRAT, il n'y a qu'un seul pivot sur un chemin source-destination, mais le choix des pivots n'est pas limité au voisinage de la source. Dans [LJK07], les auteurs proposent un protocole de routage avec pivots qui contribue à réduire le nombre de messages de contrôle et étendre la durée de vie du réseau. Les nœuds pivots sont déterminés comme suit. La destination propage une requête et sélectionne des nœuds candidats comme des pivots en se basant sur la distance. Un nœud est candidat si sa distance à la destination (ou au pivot précédent) dépasse un seuil. Chaque nœud maintient un chemin de retour au pivot précédent (ou bien à la destination). En outre, plusieurs chemins peuvent être maintenus entre les pivots. PiRAT utilise lui un seul pivot par chemin. Notre objectif est de sélectionner des pivots de sorte que les chemins des sources à la destination se chevauchent peu pour limiter les risques de congestion.

Sélection des pivots

La performance de PiRAT est principalement liée à la sélection des nœuds pivots. Les nœuds pivots ne doivent pas être sur le chemin le plus court reliant la source à la destination, pour éviter que PiRAT ne se comporte comme le protocole de routage raccourci de l'arbre. Cependant, les pivots ne doivent pas être trop éloignés du plus court chemin, afin de ne pas augmenter inutilement le nombre de sauts qu'une alarme doit traverser pour arriver à la destination. En outre, les pivots ne doivent pas être trop proches de la source pour s'assurer que les chemins ne convergent pas trop tôt.

Dans l'annexe A, nous présentons un algorithme optimal pour sélectionner les chemins. Cet algorithme nécessite une connaissance globale de la topologie et des conditions de propagation. En outre, il est centralisé et requiert beaucoup de ressources de calculs, ce qui le rend irréaliste pour un déploiement réel dans un réseau. Nous proposons donc une heuristique d'algorithme distribué qui ne nécessite qu'une connaissance locale. Enfin, nous comparons le comportement de l'algorithme optimal avec le comportement de notre algorithme distribué, afin de le valider.

Dans la suite de ce chapitre, nous décrivons la heuristique de la sélection des pivots et nous la validons.

Heuristique pour la sélection des pivots. En pratique, il n'est pas réaliste de chercher l'ensemble des chemins optimaux reliant les sources à la destination en utilisant le programme ILP (pour *Integer Linear Programming*) détaillé dans l'annexe A. En outre, cette formulation optimale suggère des hypothèses non réalistes sur le réseau (comme la stabilité des conditions de propagation) et sur les nœuds (comme la capacité de stockage et de calcul). C'est pourquoi nous proposons ici une approche heuristique basée sur un algorithme distribué qui ne nécessite que des connaissances locales.

Nous proposons de sélectionner les pivots dans une grande zone afin d'éviter la convergence des chemins, et donc permettre d'équilibrer la charge du trafic dans le réseau. Un nœud est considéré comme un pivot s'il répond aux critères suivants :

1. il est plus proche de la destination que de la source,
2. il n'est pas situé sur le plus court chemin reliant la source à la destination,
3. il n'est pas situé dans une zone à faible densité.

La première condition impose que le pivot soit plus proche de la destination que de la source. Ceci est nécessaire pour éloigner de la source le point de convergence des chemins. La deuxième

condition impose que le chemin passant par le pivot ne suit pas le plus court chemin de la source à la destination, qui est la zone du réseau la plus encombrée quand les alarmes sont générées. La troisième condition est nécessaire car les nœuds situés dans des zones à faible densité ont moins d'options de routage que les autres nœuds, ce qui réduit la diversité de routage disponible et peut augmenter la congestion.

La distance utilisée dans les conditions précédentes devrait idéalement être la plus courte distance entre les nœuds. Cependant, puisque les nœuds n'ont aucune connaissance de leur plus courte distance à la destination (ce qui nécessiterait un protocole de routage complexe), nous proposons de considérer, comme distance, le nombre de sauts qu'un paquet parcourt selon le protocole de routage raccourci de l'arbre. Cette métrique est plus précise que le nombre de sauts calculé selon le protocole de routage hiérarchique, puisqu'elle est basée à la fois sur l'environnement (à travers la table des voisins) et sur la topologie de l'arbre.

La figure 5.13 représente les trois étapes nécessaires pour sélectionner un pivot. Tout d'abord, nous considérons que les nœuds localisés dans les zones susceptibles d'envoyer des alarmes savent *a priori* qu'ils sont des sources potentielles. Ces nœuds initient une phase de découverte de pivots en diffusant un message de découverte des pivots appelé PDM *(Pivot Discovery Message)* (cf. figure 5.13(a)). Ce message contient la distance entre la source s et la destination d (calculée selon le protocole de routage raccourci de l'arbre). Quand un nœud n reçoit un PDM, il vérifie que les trois conditions suivantes sont vraies :

- $d(s,n) > d(n,d)$,
- $d(s,n) + d(n,d) \geq d(s,d) + \varepsilon_1$, avec ε_1 un seuil choisi selon la taille du réseau,
- le nombre de voisins de n est plus grand qu'un seuil ε_2.

Tous les nœuds qui répondent à ces trois conditions sont des candidats pour la sélection des pivots. Chaque candidat envoie un message de notification de pivot, appelé PNM *(Pivot Notification Message)* à la source pour lui informer qu'il est un pivot potentiel (cf. figure 5.13(b)). Quand la source reçoit un certain nombre de PNM, ou bien après un certain temps, elle choisit aléatoirement un pivot parmi l'ensemble des pivots (cf. figure 5.13(c)). Si la source ne détecte aucun pivot, un nouveau message PDM est diffusé avec des seuils ε_1 et/ou ε_2 plus petits. Notons que les valeurs $\varepsilon_1 = 0$ et $\varepsilon_2 = 0$ assurent que PiRAT trouve des pivots situés sur le plus court chemin reliant la source s à la destination d.

Validation de l'heuristique. Nous comparons à présent les résultats fournis par la formulation ILP avec ceux fournis par notre heuristique.

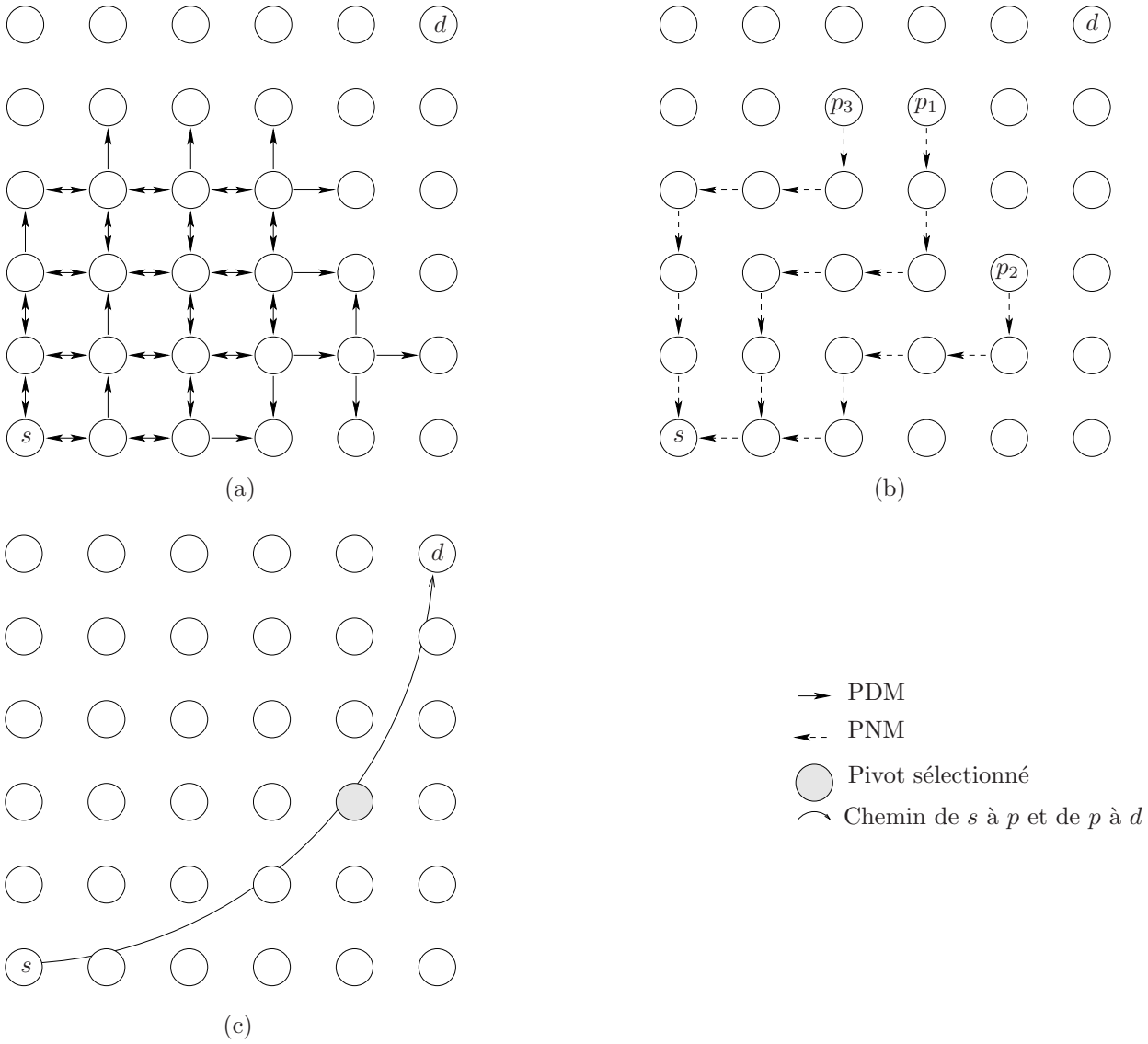


Figure 5.13 – Les trois étapes pour qu'une source s sélectionne un pivot.

Nous générons un petit réseau [3] de 36 nœuds uniformément déployés sur une surface de 60 m × 60 m. Le coordinateur du PAN est situé au centre du réseau. La portée des nœuds est fixée à 23 m. La destination est située en haut à droite du réseau (voir figure 5.14). Nous générons 10 événements indépendants (représentés par des cercles en pointillés), le rayon de chaque événement est de 13 m. Tous les nœuds appartenant à l'événement sont considérés comme des sources. Dans nos expérimentations, nous avons entre deux et trois sources pour chaque événement. Les valeurs des paramètres de l'heuristique sont les suivantes : $\varepsilon_1 = 1$ et $\varepsilon_2 = 3$.

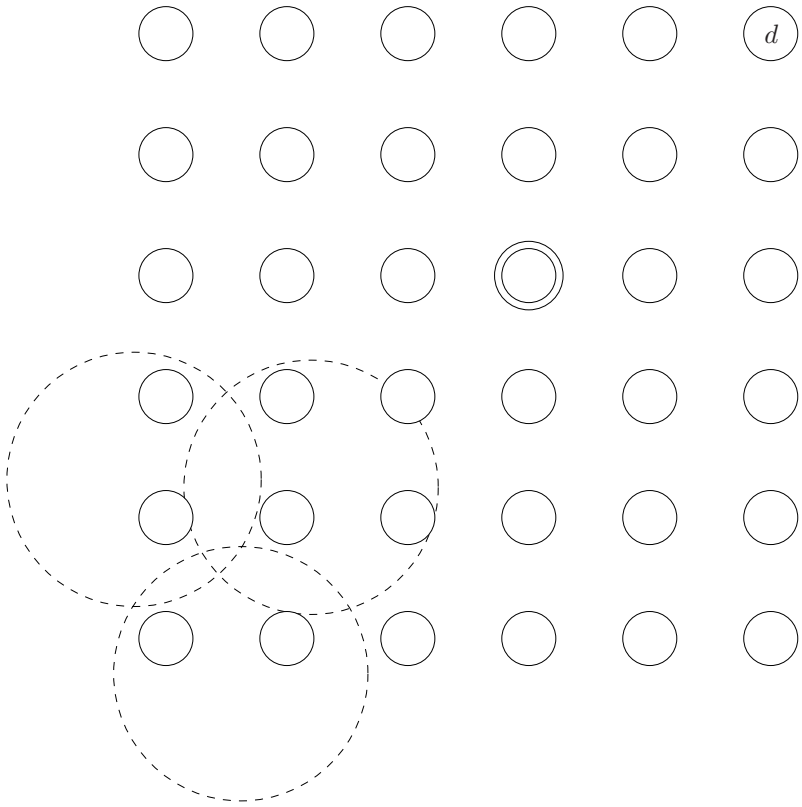


Figure 5.14 – Exemple de génération de production d'alarmes dans un réseau de 36 nœuds.

Le tableau 5.2 montre la longueur moyenne des chemins reliant la source à la destination, et le nombre moyen de liens superposés entre les différents chemins pour trois algorithmes : l'algorithme de routage par raccourci de l'arbre, nommé par la suite raccourci, l'algorithme ILP et l'heuristique. Nous remarquons que l'algorithme raccourci produit des chemins qui sont, en moyenne, 45% plus longs que ceux produits par l'algorithme ILP. Ceci est dû au fait que l'algorithme raccourci a une connaissance locale des liens du réseau alors que l'algorithme ILP calcule les plus courts chemins. Notre heuristique produit des chemins plus longs que ceux

3. Puisque nous avons décidé d'obtenir des solutions optimales en utilisant l'algorithme ILP qui s'exécute en temps non-polynomial, nous avons choisi d'utiliser un petit réseau pour nos comparaisons.

de l'algorithme raccourci puisque les pivots participent au routage. Cependant, avec la petite valeur de ε_1 que nous avons utilisée, les chemins calculés par l'heuristique sont, en moyenne, seulement 21% plus longs que ceux calculés par l'algorithme raccourci. Le nombre moyen de liens superposés est 1 pour l'algorithme ILP : pour un petit nombre de sources, le programme ILP a été capable de trouver des chemins disjoints de chaque source à la destination. Ceci est principalement dû au nombre limité de sources (qui est trois au maximum), mais ce nombre de sources est réaliste pour notre topologie. Le nombre moyen de chemins partageant des liens pour l'algorithme raccourci est $2,5$. Dans 50% des simulations, les trois chemins se superposent sur au moins un lien. Dans les autres simulations, deux chemins se superposent sur au moins un lien. Avec l'algorithme raccourci, les chemins convergent rapidement. Notre heuristique aboutit à un nombre moyen de liens superposés égal à $1,8$. La superposition des chemins n'a pas pu être complètement évitée, mais elle a pu être considérablement réduite par rapport à celle fournie par l'algorithme raccourci.

Algorithme	Longueur moyenne des chemins	Nombre moyen de liens superposés
Raccourci	3,2	2,5
ILP	2,2	1
Heuristique	3,9	1,8

Tableau 5.2 – Validation de l'heuristique de sélection des pivots.

Protocole de découverte des pivots. Comme indiqué précédemment, notre algorithme de sélection des pivots peut être implémenté en utilisant deux types de messages : les PDM et les PNM. Les PDM sont diffusés par chaque source. Chaque PDM contient les adresses de la source s et de la destination d, puisque les pivots peuvent être différents pour chaque paire (s, d). Le PDM contient de même un numéro de séquence et les paramètres ε_1 et ε_2 utilisés par l'heuristique de sélection des pivots. Quand un nœud n reçoit le premier PDM pour une paire (s, d), il détermine s'il peut être un pivot potentiel pour cette paire. S'il vérifie les trois critères indiqués dans l'heuristique, n répond, d'une manière *unicast* à la source s, par un message PNM. Si le nœud ne valide pas les conditions du choix de pivot, il rediffuse le message PDM. Les PDM supplémentaires pour la même paire (s, d) et avec un numéro de séquence inférieur ou égal à celui déjà traité sont rejetés par le nœud n. Les nœuds pivots ne retransmettent plus les PDM. Le message PNM contient la paire (s, d) et l'adresse du pivot p. Si la source s reçoit plusieurs messages PNM pour la même destination d, elle choisit aléatoirement un pivot p parmi tous les pivots potentiels. Comme nous l'avons indiqué précédemment, si une source s ne reçoit pas de

messages PNM après un certain temps, elle diffuse un nouveau message PDM avec un numéro de séquence plus grand et des valeurs de paramètres ε_1 et/ou ε_2 plus petites.

La source s peut aussi choisir un pivot p sans échanger de PDM et de PNM. Ceci a lieu quand s est incapable de trouver un pivot après plusieurs essais de paramétrage. Dans ce cas, s peut déduire un ensemble de nœuds qui sont proches de la destination d (du point de vue topologie), et les utiliser comme des pivots, en se basant sur les adresses s et d et sur les propriétés de l'adressage hiérarchique.

Nous avons implémenté et simulé notre heuristique sur des topologies où les nœuds sont uniformément distribués. Nous avons fait varier le nombre de nœuds dans le réseau de 25 à 100. Nous avons configuré trois sources choisies aléatoirement et nous avons calculé le nombre de messages PDM et PNM envoyés pour une unique destination. La figure 5.15 montre la moyenne des résultats obtenus pour 100 simulations. Nous remarquons que le nombre de messages PDM et PNM augmente quand la taille du réseau augmente. Ceci est dû au fait que, dans les grands réseaux, le nombre de nœuds participant au mécanisme de découverte des pivots augmente. Cependant, le nombre de messages n'augmente pas rapidement. Dans un réseau de 100 nœuds, seulement 200 messages PDM sont envoyés par trois sources. En effet, les nœuds pivots potentiels ne rediffusent pas les PDM. Nous remarquons aussi que le nombre de messages PNM est inférieur au nombre de messages PDM. Les messages PDM sont diffusés par chaque nœud cherchant un pivot, alors que les messages PNM suivent le chemin de chaque pivot potentiel à la source.

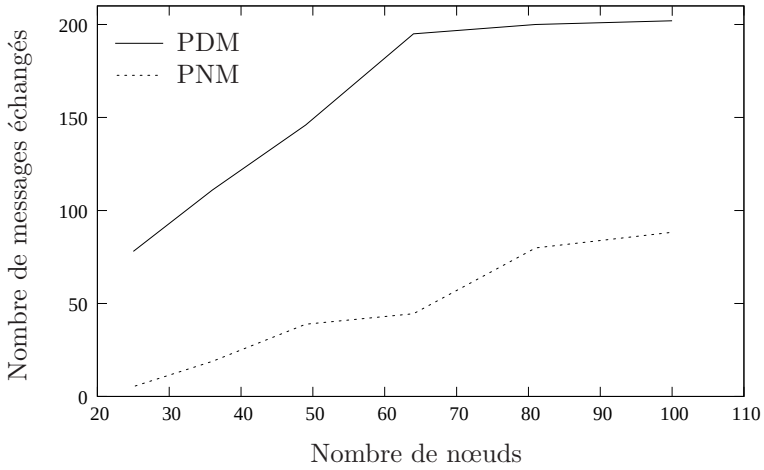


Figure 5.15 – Nombre moyen de messages de contrôle échangés.

Dans l'évaluation de PiRAT, nous considérons que les nœuds sont toujours actifs. Nous détaillons les bénéfices que PiRAT offre quand les nœuds possèdent un taux d'activité variable dans l'annexe B.

Évaluation de PiRAT

Dans cette partie, nous décrivons les simulations que nous avons effectuées afin d'évaluer PiRAT. Ici, les simulations sont lancées avec tous les nœuds activés simultanément. Nous comparons le protocole PiRAT avec le protocole de routage hiérarchique et le protocole raccourci. Nous considérons les métriques du taux de pertes de paquets, du délai de bout-en-bout, du nombre de sauts et de l'utilisation des nœuds.

Paramètres de simulation. Le simulateur utilisé est de nouveau NS2 [NS202] version 2.31. Nous avons utilisé les couches PHY et MAC du standard IEEE 802.15.4. La puissance de transmission est fixée à -25 dBm. Le modèle de propagation utilisé est le *two ray ground*. Nous avons décidé de limiter la taille des files d'attente des nœuds à 5 paquets de 34 octets (au niveau de la couche PHY), à cause des capacités de stockage limitées des composants des réseaux de capteurs sans fil. Chaque simulation est lancée pour 100 répétitions. Nous considérons une topologie simple de 100 nœuds FFD, uniformément distribués sur une surface de 100 m$\times$ 100 m. Cette topologie est illustrée sur la figure 5.16. Le coordinateur du PAN, représenté par un double cercle, est situé au centre du réseau. Les liens entre les nœuds représentent les associations père-fils. La portée d'association est fixée à 20 m. Cette valeur est plus petite que la portée de communication entre les nœuds afin de garantir que les nœuds sont associés *via* des liens de bonne qualité. Comme nous pouvons le voir sur la figure, les nœuds sont aléatoirement associés les uns aux autres. Par exemple, si le nœud 20 souhaite transmettre un paquet au nœud 50 selon le protocole de routage hiérarchique, la trajectoire du paquet passe par les nœuds suivants : $(20, 22, 23, 34, 45, 54, 52, 50)$. Les paramètres du réseau sont définis comme suit : $C_m = 5$, $R_m = 5$ et $L_m = 5$. Nous faisons varier la portée entre 30 m et 40 m.

Production du trafic d'alarmes. Le trafic d'alarmes est produit comme suit. Tout d'abord, nous considérons qu'un événement apparaît dans la partie en bas à gauche du réseau. Cet événement est détecté par tous les nœuds localisés dans le cercle pointillé. Dans nos simulations, nous considérons que l'événement est détecté dans un cercle de 25 m de rayon. Huit nœuds sont donc des nœuds sources. Notons que les événements se déclenchent après que tous les nœuds soient associés et que l'algorithme de sélection des pivots soit achevé. Il n'y a pas de trafic de base généré puisque nous concentrons notre étude sur le trafic d'alarmes seulement. Nous considérons que la destination est située en haut à droite du réseau (nœud 99). Tous les nœuds du réseau participent au routage multi-sauts. Finalement, nous considérons que les noti-

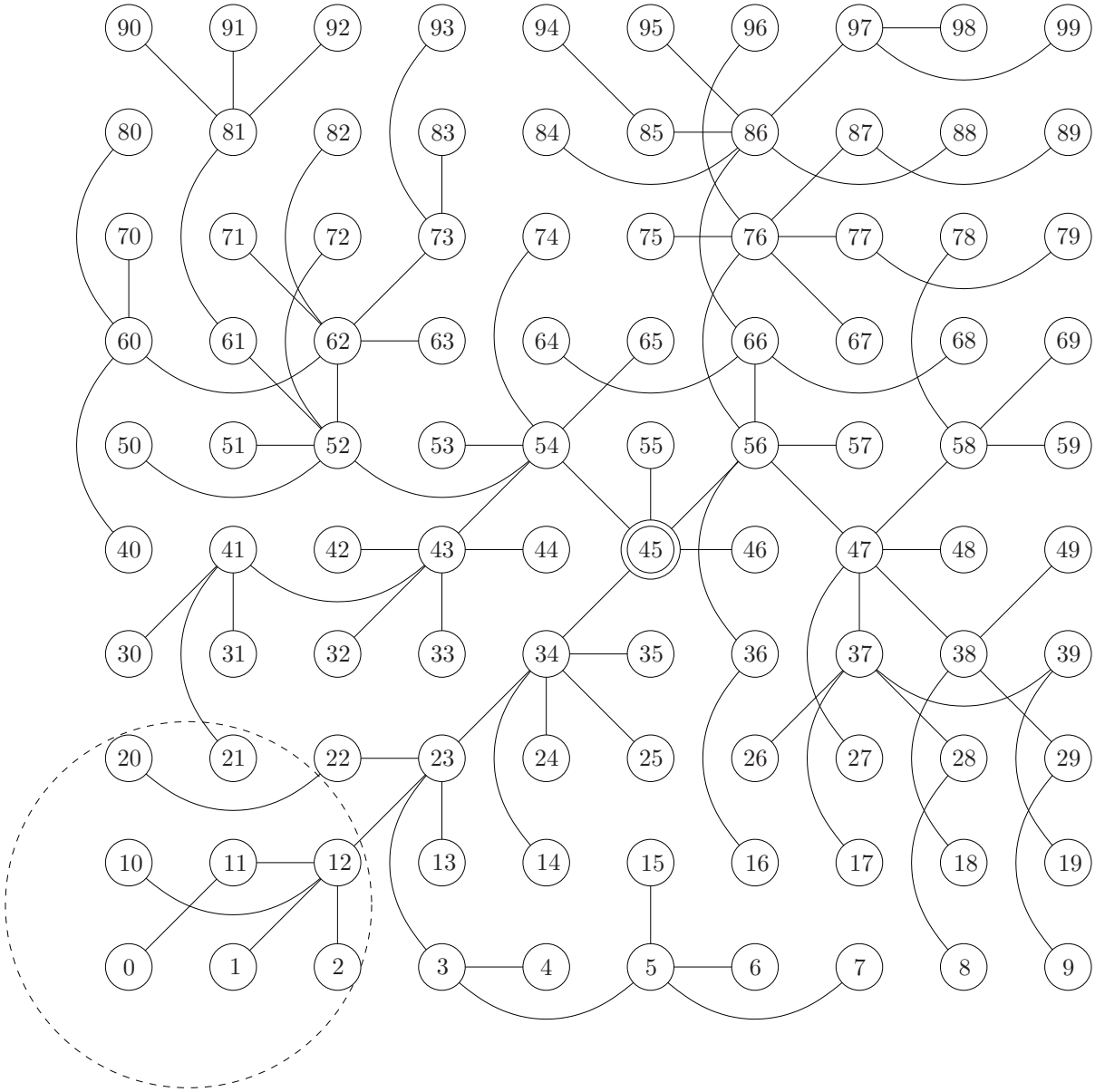


Figure 5.16 – Exemple d'une topologie réseau. Les liens représentent les associations père-fils dans le réseau.

fications des alarmes durent pour 30 secondes et nous faisons varier le taux de transmission des paquets alarmes de 1 paquet par seconde à 30 paquets par seconde. Les alarmes sont produites périodiquement pour informer la destination de l'évolution de l'événement au cours du temps.

Taux de pertes. Nous définissons le taux de pertes comme le rapport du nombre de paquets reçus avec succès par la destination sur le nombre de paquets générés par les nœuds sources. Le ratio taux de pertes prend en compte les pertes dues aux collisions et aux débordements de files d'attente.

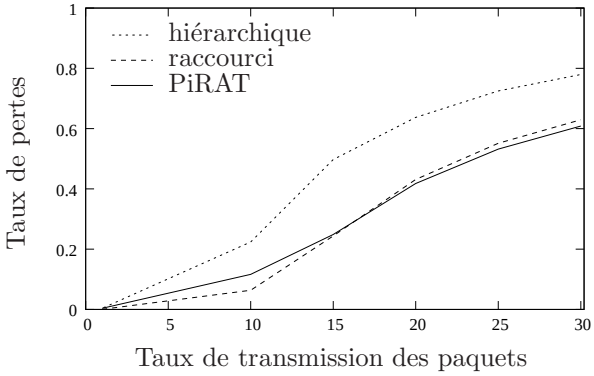


Figure 5.17 – Taux de pertes moyen pour une portée de 30 m.

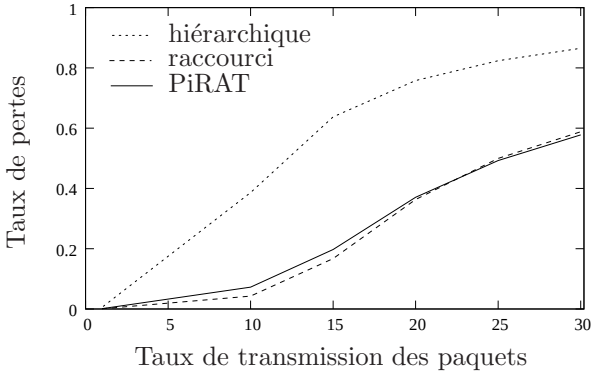


Figure 5.18 – Taux de pertes moyen pour une portée de 40 m.

La figure 5.17 et la figure 5.18 montrent le taux de pertes moyen en fonction de taux de transmission d'alarmes pour une portée de 30 m et 40 m respectivement. Comme attendu, nous remarquons que le taux de pertes augmente, pour tous les protocoles, avec le taux de transmissions des alarmes. Puisque le protocole de routage hiérarchique utilise des chemins relativement longs, la probabilité de perdre les paquets est élevée. Elle peut atteindre à 80% pour un taux de transmission de 30 paquets par seconde et une portée de 30 m, et 85% pour le même taux de transmission mais avec une portée de 40 m. Le protocole de routage raccourci et PiRAT sont capables de produire des taux de pertes plus petits que ceux du protocole de routage hiérarchique, en raccourcissant le chemin des sources à la destination. Ces deux protocoles atteignent un taux de perte de 60% pour un taux de transmission égal à 30 alarmes par seconde et pour les deux portées étudiées.

Délai de bout-en-bout. Nous définissons le délai de bout-en-bout comme l'intervalle de temps moyen séparant la génération d'un paquet par la source à la réception du même paquet par la destination. Le délai de bout-en-bout ne prend en considération que les paquets qui sont correctement reçus par la destination.

La figure 5.19 et la figure 5.20 montrent le délai de bout-en-bout moyen, en fonction du taux de transmission des paquets alarmes, et pour une portée de 30 m et de 40 m respectivement.

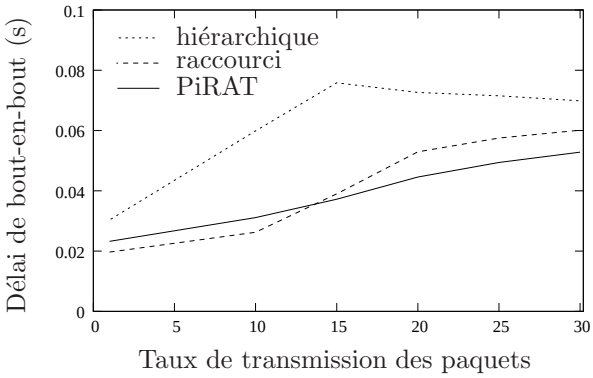


Figure 5.19 – Délai de bout-en-bout moyen pour une portée de 30 m.

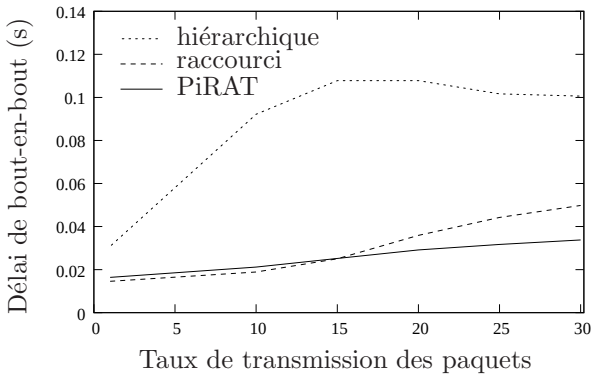


Figure 5.20 – Délai de bout-en-bout moyen pour une portée de 40 m.

Pour le protocole de routage hiérarchique, le délai augmente rapidement et devient stable après un taux de transmission égal à 15 paquets par seconde. Ceci est dû au fait que les routes sont longues et le nombre de retransmissions des paquets est élevé (puisque le taux de pertes est élevé, cf. figure 5.17 et figure 5.18). Quand le taux de transmission des alarmes augmente, le taux de pertes devient plus grand car de nombreux paquets sont rejetés. Les paquets les plus susceptibles d'être rejetés sont ceux qui correspondent à des chemins longs. Seuls les paquets qui suivent des chemins courts sont considérés pour calculer le délai de bout-en-bout, ce qui conduit à la stabilisation du délai.

Pour le protocole de routage raccourci de l'arbre et PiRAT, le délai augmente avec le taux de transmission des alarmes. Ceci est principalement dû à l'augmentation de la congestion dans le réseau et à la nécessité de retransmissions des paquets. Cependant, ces deux protocoles améliorent les performances du protocole de routage hiérarchique. Quand le taux de transmission d'alarmes est élevé, PiRAT montre le meilleur comportement en termes de délai. PiRAT réduit le délai de bout-en-bout de 28% par rapport au protocole raccourci, quand le taux de transmission d'alarmes est de 30 paquets par secondes et pour une portée de 30 m, et de 40% pour une portée de 40 m. Quand la portée augmente, le protocole de routage raccourci de l'arbre et PiRAT montrent un meilleur comportement puisque les nœuds possèdent plus de voisins pour acheminer les paquets.

Nombre de sauts. Nous définissons le nombre de sauts comme le nombre moyen de nœuds intermédiaires requis pour acheminer un paquet de la source à la destination. Seuls les paquets qui sont correctement reçus au niveau de la destination sont considérés.

La figure 5.21 et la figure 5.22 représentent le nombre de sauts en fonction du taux de transmission des alarmes, pour une portée de 30 m et 40 m respectivement. Comme prévu, le nombre de sauts est indépendant de la charge du réseau. Avec le protocole de routage hiérarchique, les paquets reçus suivent un chemin de 9 sauts en moyenne, quelle que soit la portée des nœuds. Seul le seuil de puissance conditionnant les associations entre les nœuds distant de 20 m en plus est pris en considération dans ce protocole de routage. Le protocole de routage raccourci réduit le nombre de sauts, puisqu'il route les paquets en suivant des chemins courts. Nous remarquons que la longueur moyenne des chemins est de $5,5$ sauts pour une portée de 30 m et de $4,4$ sauts pour une portée de 40 m. Quand la portée d'un nœud est grande, il peut utiliser plus de voisins pour raccourcir l'arbre. Avec PiRAT, les routes sont plus longues que celles calculées par le protocole de routage raccourci, mais notre protocole de sélection de pivots assure que le nombre de sauts n'est pas trop grand. La longueur moyenne des chemins calculées par PiRAT est de $6,5$ sauts pour une portée de 30 m et de $4,8$ sauts pour une portée de 40 m.

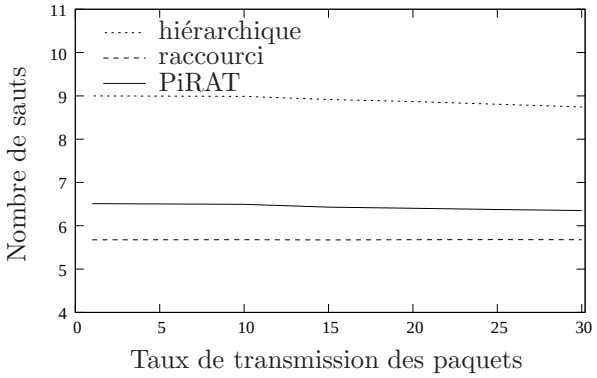


Figure 5.21 – Nombre de sauts moyen pour une portée de 30 m.

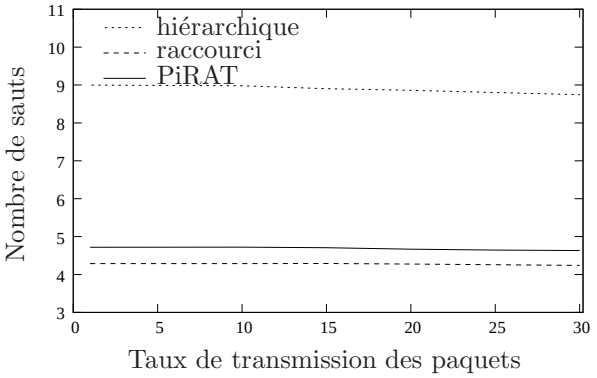


Figure 5.22 – Nombre de sauts moyen pour une portée de 40 m.

Utilisation des nœuds. L'utilisation des nœuds indique combien de nœuds participent au routage, et combien de fois ils acheminent les paquets.

Cette métrique est la métrique la plus importante pour PiRAT. PiRAT améliore le protocole de routage raccourci en introduisant de la diversité dans le routage et en diminuant les zones de congestion dans le réseau. Avec PiRAT, seule la zone aux alentours de la destination est congestionnée (voir figure 5.23 et figure 5.24).

La figure 5.23 montre la participation des nœuds au routage pour le protocole raccourci. Les sources sont les nœuds 0, 1, 2, 10, 11, 12, 20 et 21. Les lignes épaisses indiquent que le lien est fréquemment utilisé, alors que les lignes minces indiquent que le lien est rarement utilisé. Comme nous pouvons le voir, les chemins utilisés par le protocole raccourci partagent

un nombre significatif de nœuds. Le grand nombre de lignes épaisses prouve que le protocole raccourci produit de la congestion tout au long du chemin commun.

La figure 5.24 montre la participation des nœuds au routage pour PiRAT. Les nœuds pivots choisis par les sources sont représentés par des cercles grisés. Les chemins des sources à la destination évitent la région centrale du réseau afin de réduire la congestion. La nature probabiliste de PiRAT peut être identifiée par le fait que chaque nœud utilise plusieurs chemins pour arriver à la destination. Le protocole raccourci utilise 21 nœuds dans le routage, alors que PiRAT en utilise 42. Sur cet exemple, PiRAT double le nombre de nœuds qui participent au routage, et par conséquent, il réduit le nombre de paquets transmis par nœud. Ceci aboutit à réduire la surcharge des chemins et répartit la consommation d'énergie entre les nœuds. Ce phénomène augmente la durée de vie du réseau.

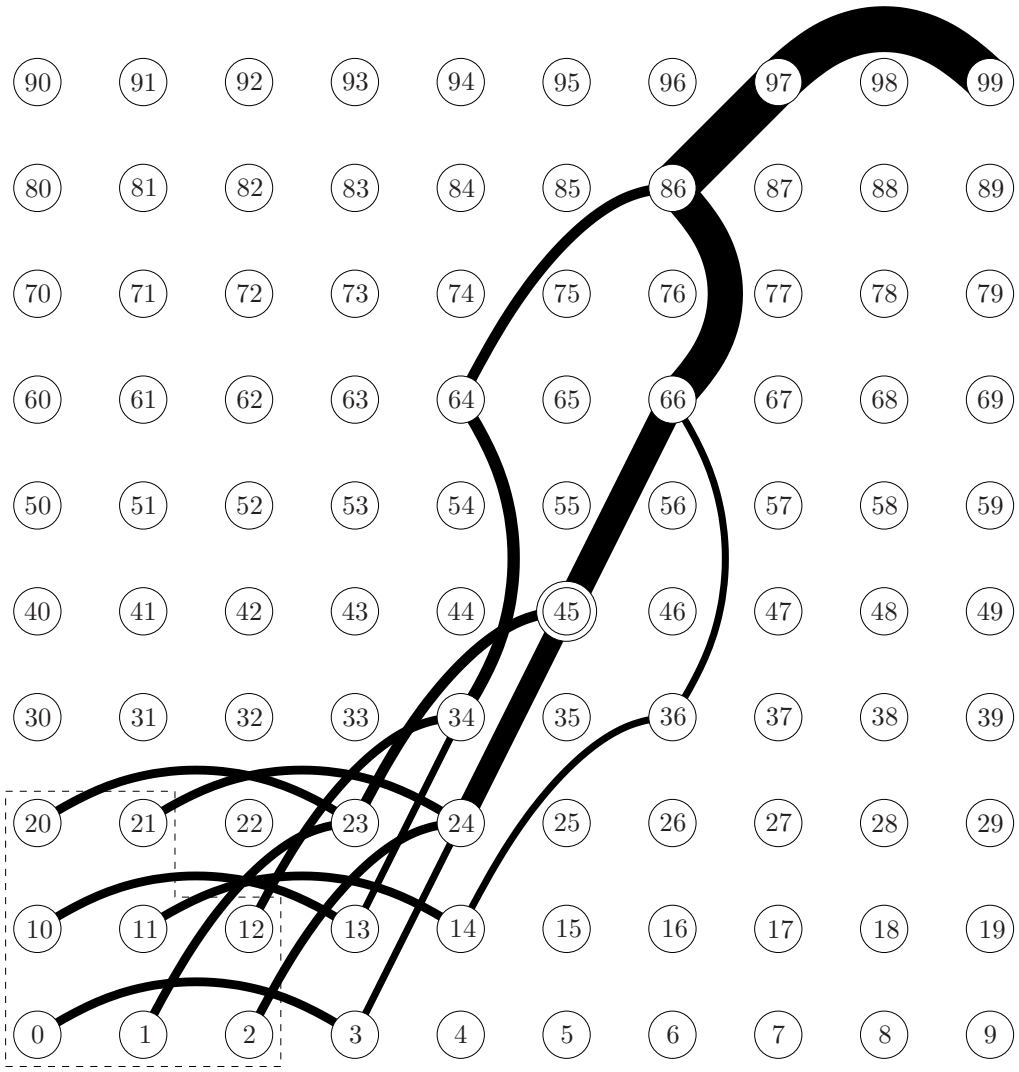


Figure 5.23 – Utilisation des nœuds dans le protocole de routage raccourci.

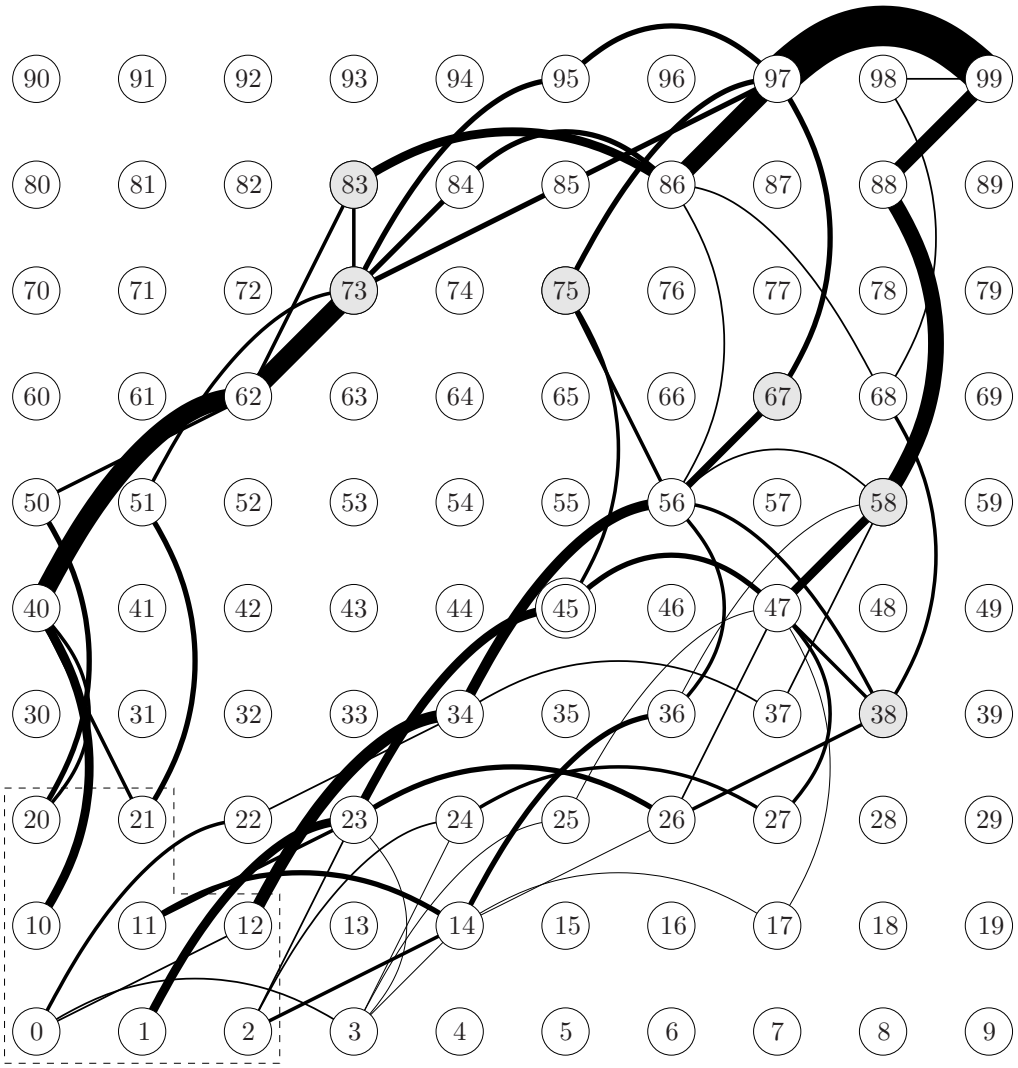


Figure 5.24 – Utilisation des nœuds dans le protocole de routage PiRAT.

5.2.2 Protocole de routage pour des réseaux multi-puits

Lorsque plusieurs sources envoient les données à une destination unique, la congestion vers la destination est inévitable.

Nous motivons tout d'abord le fait d'avoir plusieurs puits dans un réseau et l'intérêt d'utiliser des communications *anycast (one-to-any)*. Nous montrons que le fait d'avoir des chemins distants (et donc réduisant la congestion) est un problème NP-complet, et nous proposons une formulation ILP. Puisque la solution optimale n'est pas applicable dans un réseau de capteurs sans fil en raison de faibles ressources des entités, nous proposons une heuristique. Nous proposons la stratégie résultante, nommée S4, et nous la comparons avec deux autres stratégies fréquemment utilisées.

Motivations

Dans un réseau de capteurs sans fil, les nœuds collectent les données et les envoient à une station de récolte de données appelée puits. Le puits possède une grande capacité de mémoire, et n'a généralement pas de contraintes en énergie, contrairement aux nœuds. Le puits joue plusieurs rôles. Il peut stocker des données historiques, analyser les données pour détecter les situations urgentes, ou bien agir comme une passerelle assurant la connectivité avec un autre réseau filaire ou non.

Comme le trafic converge vers le puits, les nœuds proches du puits sont plus sollicités et risquent de consommer leur énergie plus que le reste du réseau. Quand tous les nœuds aux alentours du puits ont épuisé leur énergie, le puits n'est plus capable de recevoir de données, et il devient isolé. Quand cette situation se produit, le réseau est considéré déconnecté. Une solution à ce problème est de déployer plusieurs puits dans un même réseau.

Le déploiement de plusieurs puits se réfère à une architecture de réseaux sans fil où chaque puits possède les mêmes capacités fonctionnelles. Récemment, l'intérêt émerge vers des scénarios avec plusieurs puits afin d'augmenter la durée de vie d'un réseau et assurer une livraison équitable de données entre les puits [KSCK05, OE04]. Un autre avantage du déploiement de plusieurs puits est d'améliorer l'agrégation de données en réduisant le délai de communications entre les nœuds et les puits [CT07, BOV06] ou bien le coût total de la communication [KS06].

Déploiement d'un réseau à plusieurs puits. Notre but est de montrer qu'afin de minimiser les interférences et de réduire les zones de congestion entre les différents chemins, toutes les sources doivent être considérées simultanément afin d'avoir des chemins disjoints. La figure 5.25

montre une topologie de deux sources s_1 et s_2 et deux puits d_1 et d_2, avec trois stratégies de sélection du puits. Dans la partie (a) de la figure, chaque source est connectée au puits le plus proche, ce qui génère une contention aux alentours du puits d_2. Dans la partie (b) de la figure, chaque source est connectée à un puits différent afin de répartir le trafic aux puits. Cependant, le chemin (s_1, d_2) et le chemin (s_2, d_1) se croisent. La congestion est générée dans la zone où ces deux chemins se croisent. Dans la partie (c) de la figure, les chemins (s_1, d_1) et (s_2, d_2) sont distants l'un de l'autre. Le trafic parcourant le premier chemin a un impact négligeable sur le trafic parcourant le second chemin (à condition que ces deux chemins soient suffisamment distants). Cette troisième stratégie de sélection de puits ne peut être réalisée qu'en considérant les sources et les puits simultanément. Notons aussi que la sélection du puits et des chemins de routage doivent être effectuées en même temps.

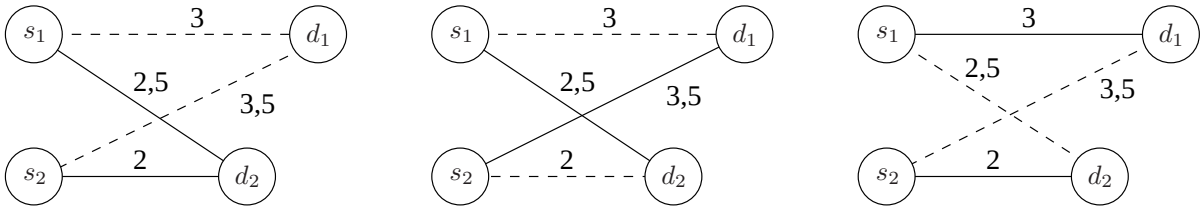


Figure 5.25 – Pour minimiser les interférences et réduire la congestion sur les chemins reliant une paire (source-puits), toutes les sources et les puits doivent être considérés simultanément, comme dans (c).

Modélisation

Dans l'annexe C, nous étudions le problème de sélection d'un puits pour chaque source. Nous étudions simultanément le problème qui consiste à trouver un chemin pour chaque source et son puits attribué, de sorte que les chemins soient distants les uns des autres pour réduire la congestion.

Dans la suite de cette partie, nous détaillons des approches heuristiques qui répondent au problème de déploiement des puits dans un réseau de capteurs sans fil.

Heuristique

Comme montré dans l'annexe C, il n'est pas envisageable de trouver des chemins toujours distants d'au moins δ sauts. En outre, la formulation optimale rend les hypothèses concernant le réseau non utilisable en pratique. Nous proposons donc dans cette partie une approche heuristique appelée stratégie de sélection simultanée de puits *(simultaneous sink selection strategy)*,

combinée avec un routage basé sur des pivots. Avant de décrire cette nouvelle stratégie, nous présentons deux stratégies utilisées fréquemment.

RSSS : *Random Sink Selection Strategy***.** Dans RSSS, les puits sont aléatoirement choisis [SM05]. Les paquets sont routés en utilisant le chemin le plus court reliant la source au puits sélectionné. Cette stratégie peut donc conduire à une congestion importante dans plusieurs zones du réseau.

La figure 5.26 montre un exemple de stratégie RSSS. Nous considérons un exemple simple de réseau avec trois sources (s_1, s_2, s_3) et trois puits (d_1, d_2, d_3). À chaque source est associée un puits aléatoirement. Les chemins entre les paires (source-puits) ne sont pas suffisamment distants et quelques nœuds interviennent dans plus d'un chemin. Ceci aboutit à une zone de congestion relativement importante.

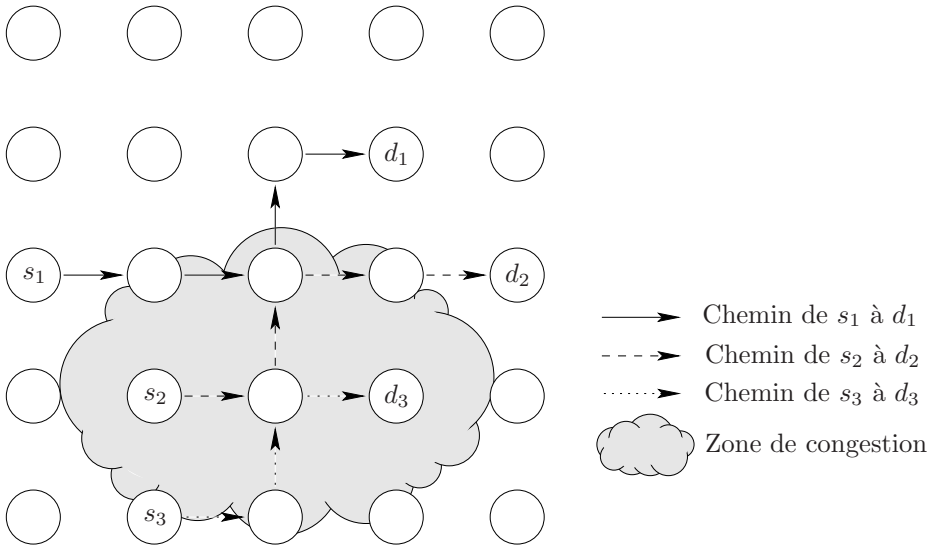


Figure 5.26 – Congestion causée par la stratégie RSSS.

CSSS : *Closest Sink Selection Strategy***.** Dans CSSS, chaque source est connectée à son plus proche puits [LZY08]. La distance de la source à chaque puits peut être calculée en utilisant le nombre de sauts. CSSS ne prend pas en considération le fait que les zones entourant les puits peuvent être congestionnées.

La figure 5.27 montre un exemple de stratégie CSSS. Chaque source choisit le puits qui lui est le plus proche. Dans l'exemple, les trois sources choisissent le même puits d_3. Nous pouvons remarquer que cette stratégie aboutit naturellement à une zone de congestion aux alentours du puits d_3.

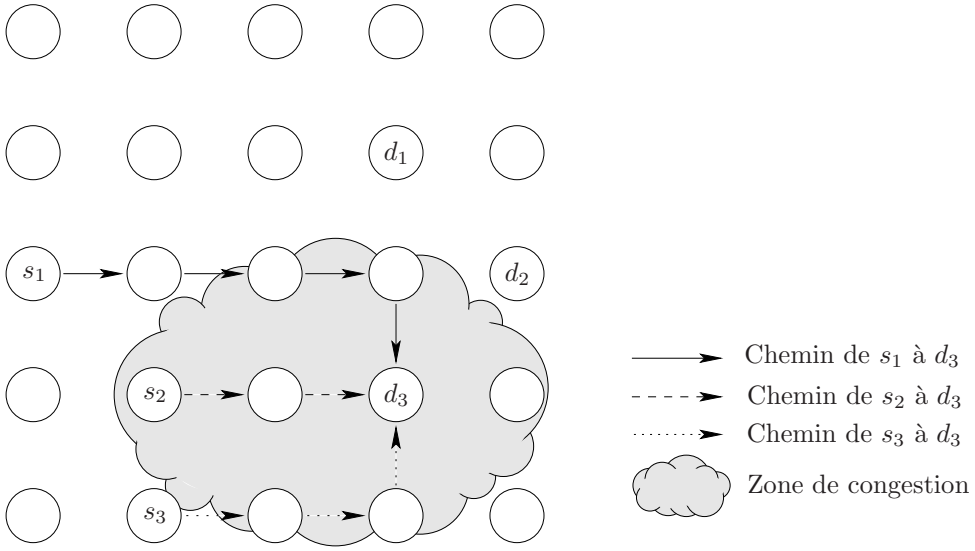


Figure 5.27 – Congestion causée par la stratégie CSSS.

S4 : *Simultaneous Sink Selection Strategy.* S4 [ERGM10] que nous proposons est une évolution de PiRAT. S4 utilise un algorithme gourmand pour sélectionner les puits et calculer les chemins. Chaque source est considérée séquentiellement. Pour chaque source, un nœud pivot est sélectionné afin de rendre les chemins les plus disjoints possible. La sélection des pivots se fait comme suit. En considérant une source s, S4 considère à tour de rôle tous les nœuds comme des pivots potentiels, et tous les puits comme des destinations potentielles. Pour chaque pivot potentiel x et destination potentielle d, S4 détermine le nombre de sauts $h(s,x)$ entre s et x, et le nombre de sauts $h(x,d)$ entre x et d. S4 détermine aussi le nombre de nœuds en commun entre le chemin $p(s,x,d)$ de s à d en passant par x et $\mathcal{P}$ qui est l'union de tous les chemins déjà choisis par S4 pour les sources précédentes. Pour une source donnée s, S4 possède plusieurs chemins candidats et choisit le chemin qui minimise le nombre de nœuds en commun avec $\mathcal{P}$. S'il reste plusieurs chemins candidats, S4 choisit le chemin de plus courte distance. Notons que S4 combine la stratégie de sélection de puits avec un mécanisme de routage.

Le nombre de sauts entre deux nœuds peut être calculé en utilisant un protocole de signalement spécifique, ou en utilisant les propriétés des adresses hiérarchiques (comme celles utilisées dans le standard ZigBee par exemple). Le nombre de nœuds en commun entre deux chemins peut être calculé de la manière suivante : s envoie un premier message à x, et un second message à d *via* x. Chaque nœud sur les deux chemins peut envoyer une notification à s, qui sera par la suite capable de compter le nombre de nœuds en commun.

La figure 5.28 montre un exemple du fonctionnement de S4. S4 choisit les pivots de façon à avoir des chemins qui sont suffisamment distants pour réduire la congestion et les interférences.

Les chemins entre chaque source et son puits ne sont pas nécessairement les plus courts chemins, mais sont éloignés (lorsque c'est possible).

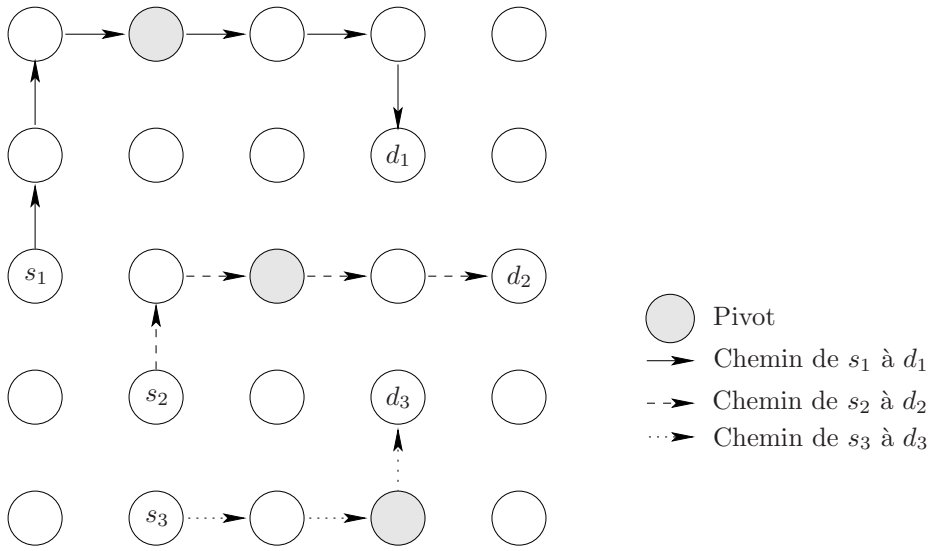


Figure 5.28 – La stratégie S4 tente d'éviter les zones de congestion du réseau en utilisant des pivots.

Pour le bon fonctionnement de S4, nous considérons qu'il existe une entité centrale qui connaît la topologie du réseau et calcule les chemins pour toutes les sources. Pour chaque source s, cette entité doit calculer le plus court chemin de s à tous les nœuds, et le plus court chemin de chaque puits jusqu'à tous les nœuds. Le premier calcul requiert $\mathcal{O}(n+m)$ opérations, où n est le nombre de nœuds et m le nombre d'arêtes. Le second calcul requiert $\mathcal{O}(n+m)$ également (notons qu'un seul calcul est lancé pour tous les puits simultanément). La complexité totale est donc $\mathcal{O}(|S|(n+m)|D|)$. Puisque $|S|$ et $|D|$ sont supposés être relativement petits, la charge de l'entité centrale dans S4 est raisonnable.

Évaluation de S4 : *Simultaneous Sink Slection Strategy*

Nous comparons S4 avec RSSS et CSSS par simulation. Nous étudions les métriques du taux de pertes et du délai de bout-en-bout.

Paramètres de simulation. Nous utilisons le simulateur NS-2, version 2.31. Nous utilisons les couches PHY et MAC en mode avec suivi de balise du standard IEEE 802.15.4. Le modèle de propagation utilisé est le *two ray ground* (avec les paramètres par défaut). La puissance de transmission est paramétrée à une valeur de -25 dBm, ce qui correspond à une portée d'environ 25 m. Les résultats obtenus sont la moyenne de 100 simulations. Dans nos simulations, nous

considérons pour des raisons de simplicité un ensemble de 49 nœuds uniformément répartis sur une surface de 70 m × 70 m. Chaque nœud est distant de 10 m de ses voisins. Tous les nœuds sont des FFD. Le coordinateur du PAN est situé au centre de la zone. Nous générons des paquets à transmettre une fois que le réseau est complètement opérationnel. Les paquets sont générés durant 50 secondes avec un taux de 2 paquets par seconde et par source. Le protocole de routage que nous utilisons pour les trois stratégies est AODV [4]. Notons qu'avant qu'AODV ne puisse envoyer des paquets à un puits inconnu, il doit établir un chemin en se basant sur des messages de contrôle, ce qui aboutit à des délais importants [5].

Taux de pertes. La figure 5.29 et la figure 5.30 présentent le taux de pertes de paquets en fonction du nombre de sources et de puits, respectivement. Nous notons que le taux de pertes de paquets pour les trois stratégies augmente toujours avec le nombre de sources, et diminue quand le nombre de puits augmente. Quand le nombre de sources dans le réseau est grand, la charge de trafic est grande et le médium est surchargé par les paquets générés. S4 est capable de réduire significativement le taux de pertes de paquets par rapport aux stratégies RSSS et CSSS. En effet, S4 vise à établir des chemins distants en sélectionnant des pivots pour chaque paire source-puits, ce qui contribue à répartir le trafic entre les nœuds et à réduire la congestion. S4 réduit le taux de pertes d'environ 36% par rapport à RSSS et à CSSS pour un nombre de sources et un nombre de puits égal à 5. S4 est plus performant que les deux autres stratégies quand le nombre de puits est égal à 1 : S4 réduit le taux de pertes d'environ 41% par rapport à RSSS et à CSSS pour cinq sources et un seul puits.

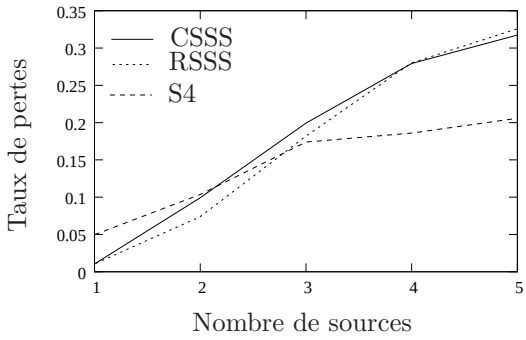


Figure 5.29 – Taux de pertes en fonction du nombre de sources, avec un nombre de puits égal au nombre de source.

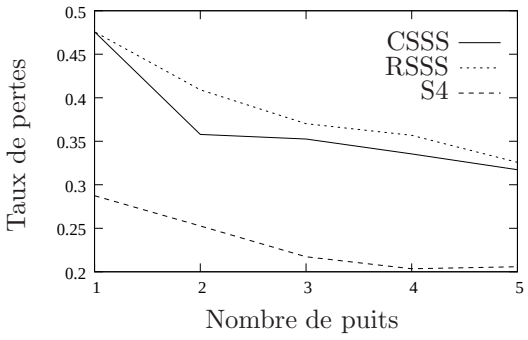


Figure 5.30 – Taux de pertes en fonction du nombre de puits, avec un nombre de sources égal à 5.

4. AODV est légèrement modifié dans S4 afin de permettre de sélectionner des pivots.

5. Pour RSSS et CSSS, AODV doit établir des chemins de chaque source à sa destination correspondante. Pour S4, AODV doit établir des chemins de chaque source à son pivot, et de chaque pivot choisi au puits correspondant.

Délai de bout-en-bout. La figure 5.31 montre le délai de bout-en-bout moyen pour les trois stratégies en fonction du nombre de sources. Pour RSSS, le délai augmente avec le nombre de sources et de puits, et devient stable quand le nombre de sources dépasse 4. Le délai diminue avec le nombre de puits (puisque la distance moyenne entre les sources et les puits diminue) et augmente avec le trafic (puisque le médium devient congestionné). Nous remarquons que CSSS produit un délai plus élevé que celui produit par RSSS. Ceci s'explique par le fait que CSSS tend à sélectionner le même puits pour plusieurs sources proches les unes des autres, ce qui contribue à des zones de congestion autour des puits. RSSS répartit l'utilisation des puits en les choisissant aléatoirement. Alors que CSSS et RSSS conduisent à des taux de pertes de paquets similaires, l'impact de la stratégie sur le délai est significatif : les paquets de grands délais sont plus susceptibles d'être rejetés en utilisant RSSS, ce qui réduit le délai de bout-en-bout moyen pour cette stratégie. S4 montre le meilleur comportement parmi les trois stratégies. Ceci prouve qu'il est important de considérer toutes les sources simultanément pour la sélection des puits, et pour établir des chemins distants durant le processus de routage. S4 réduit le délai de bout-en-bout moyen de 70% par rapport à la stratégie CSSS, et de 50% par rapport à la stratégie RSSS, pour un réseau de cinq sources et cinq puits.

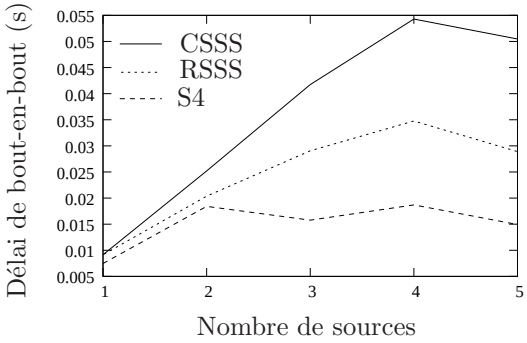


Figure 5.31 – Délai de bout-en-bout moyen en fonction du nombre de sources et de puits.

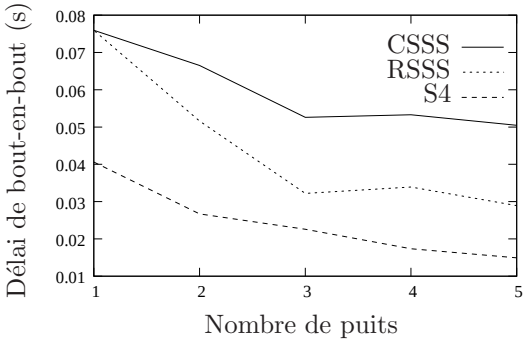


Figure 5.32 – Délai de bout-en-bout moyen avec 5 sources, en fonction du nombre de puits.

La figure 5.32 montre le délai de bout-en-bout moyen pour les trois stratégies, pour cinq sources, en fonction du nombre de puits. Pour les trois stratégies, nous remarquons que le délai diminue quand le nombre de puits augmente. Avec un grand nombre de puits, moins de congestion a lieu dans le réseau, et le nombre de retransmissions de paquets diminue (puisque le taux de pertes diminue aussi, voir figure 5.30). S4 est plus performant que les deux autres stratégies, même avec un seul puits dans le réseau. Avec un puits, S4 réduit le délai de 47% par rapport à CSSS et à RSSS.

5.2.3 Synthèse

Dans la partie 5.2 de ce chapitre, nous avons montré que la congestion est un problème important lors de la détection d'un événement urgent dans un réseau de capteurs sans fils. Nous avons proposé un nouveau protocole de routage, PiRAT, qui consiste à utiliser des nœuds pivots afin d'ajouter de la diversité au processus de routage. Les résultats de simulation ont montré que PiRAT est plus performant que les protocoles de routage réactifs et que les protocoles de routage proactifs déterministes en terme de délai, sans influencer négativement le taux de pertes. De plus, PiRAT montre une meilleure performance en terme de congestion puisqu'il utilise un grand nombre de nœuds pour le routage des paquets de la source à la destination.

Cependant, PiRAT induit toujours une zone de congestion autour du puits. Pour cela, nous avons proposé une nouvelle stratégie de sélection de plusieurs puits, appelée S4. S4 est une approche centralisée qui sélectionne les puits et des pivots afin de fournir des chemins distants et de réduire la congestion au niveau des puits. Les résultats de simulation ont montré que S4 présente de bonnes performances en termes de délai et de taux de pertes.

5.3 Conclusion

Dans ce chapitre, nous avons montré que la congestion est un problème important dans le réseau. Nous avons montré que la congestion lors du déploiement du réseau empêche l'installation de réseaux avec un grand nombre de nœuds. Pour réduire ce problème, nous avons proposé deux mécanismes de déploiement : le premier, SNAIL, est convenable pour des réseaux de capteurs linéaires, et le deuxième, *Bull's Eye*, est convenable pour des réseaux de capteurs non linéaires. Ces deux solutions permettent l'installation de grands réseaux.

Ensuite, nous avons abordé le problème de congestion durant la phase de communications entre les nœuds du réseau. Nous avons montré que les protocoles de routage réactifs, comme AODV, ne sont pas convenables pour les réseaux de capteurs sans fil vu les délais qu'ils imposent aux paquets. De même, nous avons montré qu'avec les protocoles de routage proactifs déterministes, les chemins des sources vers les destinations convergent très rapidement et contribuent à des chemins congestionnés. Nous avons proposé le protocole PiRAT, qui est une solution basée sur des pivots. PiRAT assure la diversité des routes, et donc réduit l'existence de chemins convergeant rapidement avant d'arriver à la destination. PiRAT est donc une technique de routage permettant de réduire la contention du médium localement, ce qui allège la charge de la sous-couche MAC. Ainsi, PiRAT peut être vu comme une approche de type *cross-layer*.

Cependant, PiRAT reste incapable de réduire la congestion au niveau de la destination. Une amélioration de PiRAT est le protocole S4. Avec S4, nous réalisons simultanément une diversité de routes et une diversité de puits. À nouveau, cette approche est de type *cross-layer* puisqu'elle vise à réduire la contention au niveau des nœuds et surtout des puits.

Conclusion

Les réseaux de capteurs sans fil sont utilisés de nos jours dans plusieurs applications afin de surveiller l'environnement et de détecter des événements critiques. La tendance actuelle est de déployer un unique réseau de capteurs, capable d'être utilisé par plusieurs applications simultanément, et pour lequel chaque application génère plusieurs types de trafic spécifiques, associés à différents besoins de QoS.

Les techniques de *cross-layering* sont de plus en plus utilisées dans les réseaux de capteurs sans fil pour leur capacité à garantir de meilleures performances tout en respectant la contrainte d'économie d'énergie. De plus, les techniques de *cross-layering* peuvent être utilisées conjointement avec les architectures multi-couches afin de fournir plusieurs QoS. Dans cette thèse, nous nous concentrons principalement sur le *cross-layering* entre la sous-couche MAC et la couche réseau. Nous appliquons cette technique au protocole MaCARI pour augmenter ses performances.

Contributions

Dans cette thèse, nous avons cherché à optimiser la méthode d'accès MaCARI. Pour cela, nous avons généralisé l'un des concepts sur lequel ce protocole se base en proposant une architecture multi-couches. Dans cette architecture, le temps est divisé en intervalles de temps. Pendant chaque intervalle de temps, une combinaison d'un protocole MAC et d'un protocole

de routage est activée. Cette architecture a pour but d'assurer plusieurs QoS pour plusieurs applications dans un réseau de capteurs sans fil unique. Cependant, notre architecture souffre d'un problème de dimensionnement des intervalles de temps qui influe négativement sur le débit et sur le temps de transit de paquets.

Pour répondre au problème de dimensionnement, nous avons proposé une technique de *cross-layering*. Cette technique permet des échanges entre les files d'attente $\mathcal{Q}_i$ associées à chaque combinaison $(\mathcal{M}_i, \mathcal{R}_i)$ d'un protocole MAC $\mathcal{M}_i$ et d'un protocole de routage $\mathcal{R}_i$, lorsque c'est possible. Le temps est donc réutilisé et le débit de transmission est augmenté. Pour ce faire, une combinaison de protocoles $(\mathcal{M}_i, \mathcal{R}_i)$ est autorisée à traiter les paquets des files d'attente des autres périodes durant l'intervalle de temps qui lui est associée, si la durée restante le permet et une fois que les paquets de sa propre file d'attente ont tous été traités.

Pour évaluer les échanges de files d'attente, nous avons utilisé une démarche en deux temps. Premièrement, nous avons étudié le chemin d'un paquet en considérant des hypothèses simplificatrices sur le réseau. Ces hypothèses, correspondant au pire des cas, nous ont permis d'identifier que le risque de détours est limité. Deuxièmement, nous avons étudié les performances d'un réseau plus réaliste, en considérant à la fois des files d'attente de taille limitée et une méthode d'accès réelle. Pour cela, nous avons intégré le mécanisme d'échanges de files d'attente à la méthode d'accès MaCARI. Nous avons testé l'échange avec un seul type de trafic, puis avec deux types de trafic. Dans les deux cas, nous avons montré qu'avec ce mécanisme d'échanges de files d'attente, les performances du réseau sont améliorées par rapport à celles que fournit la version MaCARI de base.

Nous avons aussi proposé d'accélérer le temps de déploiement des réseaux, et de réduire la congestion dans le réseau. Nous avons tout d'abord fourni des solutions de déploiement rapide de grands réseaux. Pour cela, nous avons proposé le mécanisme SNAIL pour le déploiement de réseaux linéaires et le mécanisme *Bull's Eye* pour les réseaux non linéaires. Les résultats de simulation ont montré que ces deux mécanismes réduisent le temps de déploiement du réseau et parviennent à associer plus de nœuds que les mécanismes traditionnels. Ensuite, nous avons abordé le problème de congestion dans la phase de communications du réseau. Nous avons proposé le protocole de routage PiRAT, un protocole de routage proactif probabiliste, basé sur des pivots. Ce protocole fournit une diversité de routage et contribue à réduire la congestion dans le réseau, sauf au niveau du puits. Le protocole S4, quant-à-lui, réduit les zones de congestion dans le réseau au niveau du puits, en considérant les sources et les puits conjointement.

Perspectives

À l'issue de ces travaux, nous tenons à présenter une brève discussion sur plusieurs points de recherche qui semblent intéressants à étudier dans l'avenir.

Changement de stratégie d'échange dans MaCARI et OCARI

Dans le mécanisme d'échanges entre files d'attente, quand une combinaison de protocoles réussit à envoyer toutes les trames de sa file d'attente, elle peut profiter du temps qu'il lui reste pour prendre en charge les trames des autres files d'attente. Une perspective est de changer cette stratégie d'échange. Selon le type de trafic et selon la priorité de l'envoi, une combinaison de protocoles MAC et routage pourrait décider d'envoyer en priorité des trames qui ne sont pas nécessairement traitées dans la période associée à cette combinaison. Cette stratégie pourrait permettre au trafic urgent (par exemple, les alarmes) d'être traité d'une manière plus rapide, ou au trafic d'archivage de données historiques (permettant, par exemple, la prévision de tendance pour les données environnementales) d'être reçu avec un taux de pertes faible. La gestion d'échange des files d'attente devient plus complexe avec cette nouvelle stratégie, mais peut assurer de meilleures QoS.

Une autre étude à mener est d'implémenter les échanges des files d'attente dans la pile OCARI, en y intégrant le mécanisme de coloriage SERENA et le protocole de routage économe en énergie EOLSR. Les protocoles de routage utilisés (le protocole de routage hiérarchique et EOLSR) risquent de causer des détours dans le réseau. Une solution est de rendre les protocoles de routage d'OCARI combinables. Une fois que le risque d'apparition de détours sera annulé, il sera intéressant de quantifier les gains du réseau en appliquant les échanges de files d'attente, et en les comparant aux gains fournis par la version OCARI de base.

Cross-layering MAC-réseau en tenant compte de la topologie et de la configuration du réseau

La technique de *cross-layering* entre la sous-couche MAC et la couche réseau est une technique très intéressante. Nous avons illustré ce fait *via* la méthode d'accès MaCARI. Néanmoins pour en évaluer les bénéfices, il faut faire l'hypothèse d'une topologie et de profils de trafic. Il serait judicieux d'étudier conjointement le *cross-layering* MAC-routage et la topologie (c'est-à-dire le positionnement des nœuds routeurs) pour offrir les QoS requises. En d'autres termes, une perspective serait d'étudier l'impact du déploiement des nœuds routeurs afin de garantir

une certaine QoS. Une autre perspective serait d'étudier les QoS qu'il est possible d'offrir sur un grand éventail de topologies.

Validation par prototypage

Les résultats de simulation de ce mémoire sont obtenus en utilisant un simulateur réseau. Notre but était de tester le bon fonctionnement des différents mécanismes que nous avons proposés. Une évaluation plus réaliste de nos différents mécanismes sur une plateforme matérielle permettrait de valider les résultats obtenus par simulation avec un médium réel. Toutefois, le passage à l'échelle est plus difficile à mettre en place dans ces conditions.

Amélioration des protocoles de routage PiRAT et S4

Le protocole de routage PiRAT que nous avons présenté est un protocole de routage proactif probabiliste, basé sur des pivots afin d'assurer une diversité de routage. Les pivots sont choisis selon des critères de positionnement par rapport à la source et à la destination, ainsi que sur le plus court chemin reliant la source à la destination. Puisque nous travaillons dans le domaine des réseaux de capteurs, la contrainte d'énergie est importante afin d'augmenter la durée de vie du réseau. Intégrer à PiRAT des contraintes d'économie d'énergie sur les pivots apporterait, en plus des gains de performances en termes de délai et de taux de pertes, un gain en terme de durée de vie du réseau (c'est à dire sur la durée quand un nœud du réseau épuise son énergie). Une contrainte sur la pondération d'énergie pourrait être ajoutée aux contraintes mentionnées pour sélectionner un pivot.

Dans le protocole de routage S4, nous avons pris comme hypothèses comme l'usage d'une méthode d'accès basée sur le mode suivi de balise ce qui sous-entend une synchronisation très complexe. Nous avons considéré aussi que les liens sont stables ce qui excluent toute mobilité des nœuds ou toute modification des conditions de propagation. Ce travail pourrait être considéré en affinant ces hypothèses.

Annexe A

PiRAT : Sélection des pivots

Formulation ILP. Dans cette partie, nous proposons de trouver un ensemble de chemins optimaux. Notons que nous ne parlons pas de pivots ici, puisque les pivots sont seulement utilisés dans PiRAT pour faire une approximation des chemins optimaux. Afin de trouver l'ensemble des chemins optimaux, nous proposons une formulation basée sur un programme linéaire en nombre entiers ILP *(Integer Linear Programming)*. Ce programme ILP calcule l'ensemble des chemins (un chemin par source), de sorte que chaque chemin soit court et partage un faible nombre de liens avec les autres chemins. Le compromis entre la longueur des chemins et le nombre maximum de liens partagés est donné par un paramètre α.

Les constantes de notre programme ILP sont les suivantes. V représente l'ensemble des nœuds du réseau, et *lien* représente la matrice d'adjacence du réseau. Le cardinal de V est représenté par $|V|$. $S \subset V$ est l'ensemble des sources, et $d \in V \backslash S$ est la destination. Le cardinal de S est représenté par $|S|$. $\alpha \in [0; 1]$ correspond au compromis entre la longueur des chemins et le nombre de chemins qui partagent un même lien.

Les variables que nous utilisons sont les suivantes. Le chemin $p[s, x, y]$ définit si le lien (x, y) est utilisé par le chemin de s à d ou non. $superposition[x, y]$ compte le nombre de chemins qui utilisent le lien (x, y). $maxSuperposition$ correspond au nombre maximum de chemins partageant le même lien. Finalement, $t[x, y]$ est une variable binaire temporaire utilisée pour calculer $maxSuperposition$.

La fonction d'objectif et les contraintes de notre programme ILP sont synthétisées dans la figure A.1.

Dans la fonction d'objectif de notre programme ILP, nous pondérons le nombre maximum

minimiser $maxSuperposition \cdot \alpha + \sum_{s \in S, x \in V, y \in V} p[s,x,y] \cdot (1-\alpha)$
sous les contraintes suivantes :

$$\forall s \in S, \forall x \in V, \forall y \in V, \quad p[s,x,y] \leq lien[x,y] \tag{1}$$

$$\forall s \in S, \quad \sum_{y \in V} p[s,s,y] = 1 \tag{2}$$

$$\forall s \in S, \quad \sum_{x \in V} p[s,x,d] = 1 \tag{3}$$

$$\forall s \in S, \forall x \in V \backslash \{d\}, \forall y \in V \backslash \{d\}, \quad p[s,x,y] \leq \sum_{z \in V \backslash \{x\}} p[s,y,z] \tag{4}$$

$$\forall x \in V, \forall y \in V, \quad superposition[x,y] = \sum_{s \in S} p[s,x,y] \tag{5}$$

$$\forall x \in V, \forall y \in V, \quad maxSuperposition \geq superposition[x,y] \tag{6}$$

$$\sum_{x \in V, y \in V} t[x,y] = |V|^2\text{-}1 \tag{7}$$

$$\forall x \in V, \forall y \in V, \quad maxSuperposition - t[x,y] \cdot |S| \leq superposition[x,y] \tag{8}$$

Figure A.1 – Programme ILP pour calculer les chemins.

de liens partagés par $\alpha \in [0;1]$, et la somme des longueurs totales des chemins par $1-\alpha$. La première contrainte du programme ILP impose que les chemins peuvent utiliser seulement les liens qui sont dans le réseau. Ensuite, nous définissons trois contraintes dans le but d'assurer que p représente l'ensemble de $|S|$ chemins, de chaque nœud de S jusqu'à la destination d. La cinquième contrainte calcule le nombre de chemins qui se superposent pour un lien (x,y) donné. Les trois dernières contraintes sont utilisées pour modéliser la fonction non convexe $maxSuperposition = \max_{x \in V, y \in V} superposition[x,y]$, au moyen de fonctions convexes. Pour cela, la sixième contrainte impose que $maxSuperposition$ soit plus grand ou égal à chaque $superposition[x,y]$. La septième et la huitième contrainte assurent que $maxSuperposition$ ne dépasse pas le $superposition[x,y]$ maximal. En effet, nous remarquons que puisqu'il y a seulement $|S|$ chemins, $superposition[x,y] \leq |S|$. Notons de même que seulement un $t[x,y]$ est égal à 0 (à cause de la huitième contrainte). Pour toutes les paires (x,y) telle que $t[x,y] = 1$, la neuvième contrainte devient $maxSuperposition \leq superposition[x,y] + |S|$ ce qui n'est pas contraignant. Pour la paire (x_0,y_0) telle que $t[x_0,y_0] = 0$, la neuvième contrainte devient $maxSuperposition \leq superposition[x_0,y_0]$. En d'autres termes, il faut que $maxSuperposition \geq superposition[x,y]$ pour tout lien (x,y) (à cause de la sixième contrainte), et il existe un lien (x_0,y_0) tel que $maxSuperposition \leq superposition[x_0,y_0]$. Ceci correspond à la modélisation de la valeur maximale d'un ensemble de variables.

Annexe B

PiRAT avec un taux d'activité variable

Dans cette annexe, nous détaillons les bénéfices que PiRAT offre quand les nœuds possèdent un taux d'activité variable. Nous supposons que chaque nœud est périodiquement actif puis inactif. Les périodes d'activités des nœuds sont indépendantes, comme dans [KAAVN06]. Nous considérons que chaque nœud connaît le cycle d'activité de ses voisins, ce qui signifie que chaque nœud peut estimer quand chacun de ses voisins est actif ou inactif. Ceci peut être réalisé par diffusion de l'ordonnancement de l'activité aux nœuds voisins.

En se basant sur ces hypothèses, nous adaptons la sous-couche MAC du standard IEEE 802.15.4 de la façon suivante. Quand la sous-couche MAC examine une trame dans la file d'attente, elle vérifie si le prochain saut de cette trame est actif ou non. S'il est actif, la sous-couche MAC envoie la trame à ce voisin. Sinon, la sous-couche MAC examine la trame suivante dans la file d'attente. En effet, le prochain saut de la trame suivante de la file d'attente peut être différent du prochain saut de la première trame de cette file d'attente, et il peut correspondre à un nœud actif. Si aucune trame de la file d'attente ne peut être acheminée, la sous-couche MAC attend jusqu'à ce qu'un prochain saut soit de nouveau actif. Cette approche est une technique de *cross-layering* couramment utilisée dans les réseaux de capteurs sans fil pour améliorer les performances des réseaux [AVA06, SL06, AACG+09]. Quand un événement urgent est détecté, plusieurs paquets sont générés pour la même destination. Dans le cas où un réseau utilise le protocole de routage raccourci, quand un nœud a plusieurs paquets d'alarmes dans sa file d'attente, tous ces paquets ont le même prochain saut (si la destination est unique). Donc, notre adaptation de la sous-couche MAC n'a aucun effet sur le protocole raccourci. Cependant, avec PiRAT, un nœud peut avoir dans sa file d'attente des paquets pour différents nœuds pivots.

Au lieu d'être bloqué à cause d'un prochain saut inactif pour le premier paquet, le nœud est capable d'envoyer un autre paquet pour un voisin actif.

Afin de mesurer les bénéfices de cette amélioration, nous évaluons par simulation le temps moyen qu'un nœud attend pour avoir un voisin actif. Nous faisons varier le nombre de voisins des nœuds, et nous activons les nœuds indépendemment. Chaque nœud est activé périodiquement chaque 2 secondes, avec un cycle allant de 50 à 100% du temps. Le taux d'activité des nœuds varie de $0,5$ à 1. Avec cette configuration, l'émetteur est toujours en mesure d'être actif en même temps que chacun de ses voisins, à un instant donné. Nous avons calculé le temps moyen d'attente avant l'activation du voisin (qui est 0 si les deux nœuds sont actifs en même temps). Les résultats obtenus sont la moyenne de 10000 répétitions.

La figure B.1 montre le temps moyen d'attente d'un nœud avant que l'un de ses voisins ne soit actif. Ce temps est mesuré en fonction du taux d'activité. Comme prévu, le temps moyen d'attente diminue quand le taux d'activité des nœuds augmente. Nous remarquons que le temps d'attente n'est pas négligeable puisqu'il peut atteindre $0,25$ secondes pour un taux d'activité égal à $0,5$. Le principal résultat représenté dans la figure est le gain qu'un réseau peut avoir en termes de temps d'attente quand un nœud considère deux ou trois voisins et non pas le nœud voisin correspondant au premier paquet de sa file (comme c'est le cas pour le protocole raccourci). Quand le taux d'activité est égal à $0,5$, le fait de considérer deux voisins au lieu d'un seul réduit le temps d'attente de 68%. De même, un nombre de voisins égal à trois réduit le temps d'attente de 88% par rapport à un voisin.

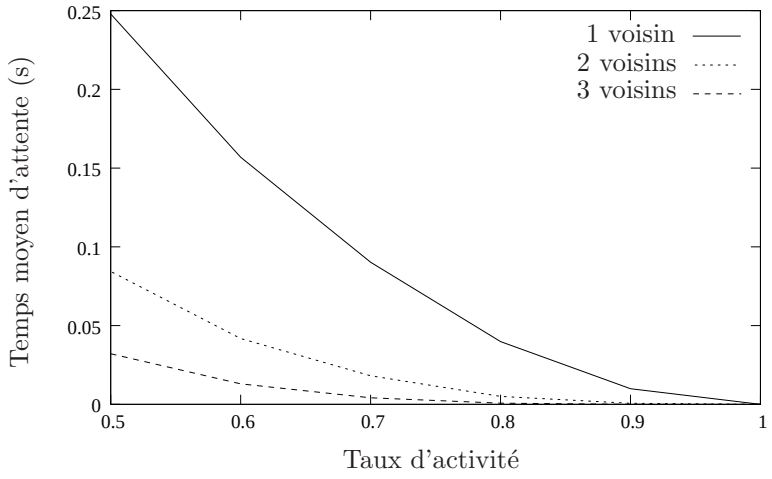


Figure B.1 – Le temps moyen d'attente est considérablement réduit quand plusieurs voisins sont considérés.

Réduire le temps moyen d'attente est critique pour un protocole de routage dans un réseau de capteurs sans fil à faible taux d'activité. En effet, chaque nœud du chemin retarde le paquet

par ce temps d'attente moyen, puisque chaque nœud doit attendre jusqu'à ce que son prochain saut soit actif. Ce délai influe négativement sur les paquets dans la file d'attente, puisqu'ils sont généralement traités dans l'ordre.

PiRAT est moins sensible à ce phénomène que d'autres protocoles de routage. Ceci est dû à deux raisons. Premièrement, puisque PiRAT utilise plusieurs nœuds intermédiaires comme pivots, les paquets de la file d'attente d'un nœud possèdent généralement des prochains sauts différents, et PiRAT peut donc envoyer n'importe lequel de ces paquets tant que le nœud correspondant au prochain saut est actif. Deuxièmement, puisque PiRAT répartit la charge du trafic sur plusieurs liens au lieu d'utiliser certains liens seulement, il peut atteindre un débit élevé même si la disponibilité du lien est réduite à cause des taux d'activité faibles des nœuds. Si la disponibilité du lien est réduite, un temps plus large est requis ce qui diminue le débit total.

Annexe C

Sélection des puits

Nous nous concentrons dans cette annexe sur la description formelle de la sélection conjointe des puits et du routage pour les communications *anycast*.

Description de la sélection conjointe du puits et du routage. Considérons un ensemble de nœuds V et un graphe $G = (V, E)$ représentant le réseau. E est défini comme suit. Si x et y peuvent communiquer entre eux, $(x, y) \in E$ et $(y, x) \in E$. Soit $S \subset V$ l'ensemble des sources, et soit $D \subset V$ l'ensemble des puits. Notons $h(x, y)$ le nombre de sauts séparant deux nœuds x et y. La distance minimale entre deux chemins p_1 et p_2, notée $h(p_1, p_2)$, peut être définie par :

$$h(p_1, p_2) = \min_{x \in p_1, y \in p_2} h(x, y).$$

Définition 3. *Le problème de la sélection du puits et du routage pour des communications sans fils* anycast*, SSRPAW (Sink Selection and Routing Problem for Anycast Wireless communications), consiste à trouver un ensemble de chemins $\{p_i\}$ qui connecte chaque source $s_i \in S$ à un puits $d_{f(i)} \in D$, tel que $h(p_{i_1}, p_{i_2}) \geq \delta$ pour tout $i_1 \neq i_2$ et pour un $\delta > 0$.*

L'objectif du problème de SSRPAW est de trouver une stratégie de sélection de puits (caractérisée par la fonction f) et une stratégie de routage (caractérisée par le choix des chemins $\{p_i\}$) qui assure que les chemins sont distants d'au moins δ sauts afin d'éviter la congestion du médium. δ dépend des conditions de propagation. Pour un réseau de grande densité, les interférences sont souvent négligeables après deux sauts, et donc, δ peut être fixé à 2.

Dans ce qui suit, nous montrons que SSRPAW est NP-complet. Ensuite, nous proposons une

formulation ILP qui permet de calculer des solutions optimales. Ensuite nous motivons le besoin d'une heuristique qui peut être réalisable avec une faible capacité de calcul et des hypothèses réalistes.

Preuve de la NP-complétude de SSRPAW. Afin de prouver la NP-complétude de SSRPAW, citons, tout d'abord, un problème similaire.

Définition 4. *Le problème d'ensemble de chemins disjoints part d'un graphe $G = (V, E)$, d'un ensemble S de k sources et d'un ensemble D de k puits. Il consiste à déterminer s'il existe k chemins nœuds-disjoints mutuellement $\{p_i\}$, tels que p_i est un chemin de s_i à $d_{f(i)}$, pour $1 \leq i \leq k$, et $f(i)$ une permutation de $\{1, \ldots, k\}$. Deux chemins disjoints mutuellement n'ont aucun nœud en commun, à l'exception de la source et de la destination.*

Théorème 5. *Le problème d'ensemble de chemins disjoints est NP-complet [QPSS94]. Il est similaire au problème des chemins nœuds-disjoints.*

Théorème 6. *SSRPAW est NP-complet pour tout δ.*

Démonstration. La preuve de NP-complétude de SSRPAW pour tout $\delta > 0$ englobe le problème d'ensemble des chemins disjoints. Nous montrons dans ce qui suit que si l'on est capable de résoudre SSRPAW pour un graphe spécifique $\bar{G}$ en un temps polynomial, on peut résoudre le problème d'ensemble de chemins disjoints dans un graphe général G en un temps polynomial (ce qui est peu probable, à moins que P = NP).

Considérons $G = (V, E)$ un graphe arbitraire. La construction de $\bar{G} = (\bar{V}, \bar{E})$ est faite comme suit. Chaque nœud $n \in V$ est aussi un nœud de $\bar{V}$. Chaque arête $e = (x, y) \in E$ devient un chemin de δ arêtes dans $\bar{E}$, connectant $x \in \bar{V}$ à $y \in \bar{V}$.

Supposons maintenant que SSRPAW peut être résolu en temps polynomial dans un graphe $\bar{G}$. Ceci veut dire qu'il existe $|S| = k$ chemins $\{\bar{p}_i\}$ dans $\bar{G}$, pour $1 \leq i \leq k$, tels que chaque chemin $\bar{p}_i$ connecte une source $s_i \in S$ à un puits $d_{f(i)} \in D$. En outre, pour tout $i_1 \neq i_2$, $h_{\bar{G}}(\bar{p}_{i_1}, \bar{p}_{i_2}) \geq \delta$, par définition de SSRPAW. Par construction de $\bar{G}$, chaque chemin $\bar{p}$ dans $\bar{G}$ peut être traduit en un chemin p dans G. Donc, nous avons k chemins $\{p_i\}$ dans G tels que chaque chemin p_i connecte une source $s_i \in S$ à un puits $d_{f(i)} \in D$. Ces chemins sont tels que $h_g(p_{i_1}, p_{i_2}) \geq \delta/\delta = 1$, ce qui veut dire qu'ils sont nœuds-disjoints. Donc, nous avons résolu le problème d'ensemble de chemins disjoints entre S et D en temps polynomial, ce qui complète la preuve. □

Formulation ILP. Le but de cette partie est de définir des solutions optimales en utilisant une formulation ILP. Cette formulation suppose que $\delta = 2$ et prend comme entrée un ensemble de nœuds V, un ensemble de sources $S \subset V$, un ensemble de puits $D \subset V$ et un ensemble de variable binaires $e_{x,y}$ représentant les arêtes. L'objectif de ce programme ILP est de trouver un ensemble de chemins $\{p_s\}$, un par source $s \in S$, tel que les chemins sont distants les uns des autres et le nombre total d'arêtes est minimisé. Chaque chemin est défini comme un ensemble de variables binaires $p_s(x, y)$, telles que $p_s(x, y)$ vaut 1 si le chemins p_s utilise l'arête (x, y), et 0 sinon. Le programme résultant est donné dans la figure C.1.

minimiser $\sum_{s\in S} \sum_{x\in V} \sum_{y\in V} p_s(x, y)$
telle que

$$\forall s \in S, x \in V, y \in V \quad p_s(x, y) \leq e_{x,y} \tag{1}$$

$$\forall s \in S \quad \sum_{y\in V} p_s(s, y) \geq 1 \tag{2}$$

$$\forall s \in S \quad \sum_{x\in V} \sum_{d\in D} p_s(x, d) \geq 1 \tag{3}$$

$$\forall s \in S, x \in V\backslash D, y \in V\backslash D \quad p_s(x, y) \leq \sum_{z\in V\backslash\{x\}} p_s(y, z) \tag{4}$$

$$\forall x \in V \quad \sum_{s\in S} \sum_{y\in V} p_s(x, y) \leq 1 \tag{5}$$

$$\forall s_1 \in S, s_2 \in S\backslash\{s_1\}, x \in V, x' \in V \quad \sum_{y\in V} p_{s_1}(x, y) + \sum_{y\in V} p_{s_2}(x, y) \leq |S| - (|S| - 1)e_{x,x'} \tag{6}$$

$$\forall s_1 \in S, s_2 \in S\backslash\{s_1\}, x \in V, x' \in V \quad \sum_{y\in V} p_{s_1}(x, y) + \sum_{y\in V} p_{s_2}(x', y) \leq |S| - (|S| - 1)e_{x,x'} \tag{7}$$

$$\forall s_1 \in S, s_2 \in S\backslash\{s_1\}, x \in V, x' \in V \quad \sum_{y\in V} p_{s_1}(x', y) + \sum_{y\in V} p_{s_2}(x, y) \leq |S| - (|S| - 1)e_{x,x'} \tag{8}$$

$$\forall s_1 \in S, s_2 \in S\backslash\{s_1\}, x \in V, x' \in V \quad \sum_{y\in V} p_{s_1}(x', y) + \sum_{y\in V} p_{s_2}(x', y) \leq |S| - (|S| - 1)e_{x,x'} \tag{9}$$

Figure C.1 – Le programme ILP résolvant SSRPAW pour $\delta = 2$.

L'inégalité (1) indique qu'il est interdit d'utiliser une arête (x, y) dans un chemin si x et y ne sont pas voisins. La contrainte (2) indique que pour toute source s, il existe au moins une arête qui parte de s dans le chemin p_s. La contrainte (3), symétriquement, indique que

chaque chemin p_s doit terminer dans un nœud appartenant à D. Cette contrainte correspond aux communications *anycast*. La contrainte (4) est une contrainte de connectivité : pour chaque arête (x, y) d'un chemin p_s, il y a au moins une arête (y, z) dans p_s (avec $z \neq x$). En d'autres termes, chaque arête (x, y) d'un chemin p_s est suivie par une arête (y, z) sur le même chemin (à l'exception de l'arête (y, d)). La contrainte (5) indique que chaque arête (x, y) est utilisée par un chemin au plus (car $\delta > 0$). Les contraintes (6), (7), (8) et (9) indiquent que la distance entre les chemins doit être au moins 2 ou plus. Plus précisément :

- Si l'arête (x, x') existe dans le graphe, il existe au plus un chemin qui utilise le nœud x ou le nœud x', puisque x et x' sont des voisins. En effet, si l'arête (x, x') existe, $e_{x,x'}$ est égale à 1, et la partie droite de l'équation est égale à $|S| - (|S| - 1) = 1$ (où $|S|$ représente le cardinal de S).

- Si l'arrête (x', x) n'existe pas dans le graphe, le nombre de chemins qui utilisent les nœuds x et x' n'est pas limité. Dans ce cas, $e_{x,x'} = 0$ et la partie droite des équations est égale à $|S|$. Puisque $|S|$ est une limite naturelle du nombre de chemins, l'inégalité ne pose pas de restriction.

Ces quatre équations ne peuvent pas être fusionnées en une seule, parce qu'il est possible qu'un même chemin p_{s_1} utilise l'arête (x, x'). Dans ce cas, la sommation peut compter deux arêtes pour ce chemin (une pour (x, x') et une pour (x', y)), et le résultat serait strictement supérieur à 1. Notons, cependant, que la contrainte (5) (correspondant à $\delta = 1$) n'est pas requise car elle est incluse dans les contraintes (6) à (9).

Calcul des solutions optimales. Dans ce paragraphe, nous étudions les solutions optimales trouvées par notre programme ILP, exécuté sous GLPK *(GNU Linear Programming Kit)*. Nous générons une topologie de 7×7 nœuds, où chaque nœud peut communiquer avec ses quatre voisins directs seulement. Nous cherchons des chemins reliant chaque source à un puits, avec une distance minimale entre les chemins δ égale à 2. Les sources et les puits sont choisis aléatoirement de telle sorte que toutes les sources soient à une distance de δ les unes des autres, et tous les puits soient à une distance de δ les uns des autres. Nous faisons varier le nombre de sources et le nombre de puits. Les résultats obtenus sont des moyennes de 50 simulations.

La figure C.2 montre que le coût moyen du chemin optimal augmente avec le nombre de sources et de puits. Quand le nombre de sources et de puits devient plus grand, les chemins

sont plus longs afin d'assurer qu'ils soient disjoints. La figure C.3 montre le pourcentage des chemins optimaux trouvés en fonction du nombre de sources et de puits. Quand le nombre de sources est plus petit que quatre, des solutions optimales existent toujours. Quand le nombre de sources est de cinq ou six, il n'existe pas toujours de solutions optimales, c'est-à-dire qu'il n'est pas possible de trouver des chemins distants de $\delta = 2$.

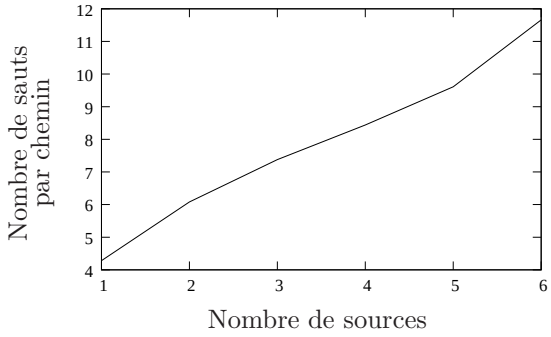


Figure C.2 – Coût moyen d'un chemin optimal par paire source-puits, en fonction du nombre de sources, avec un nombre de puits égal au nombre de sources.

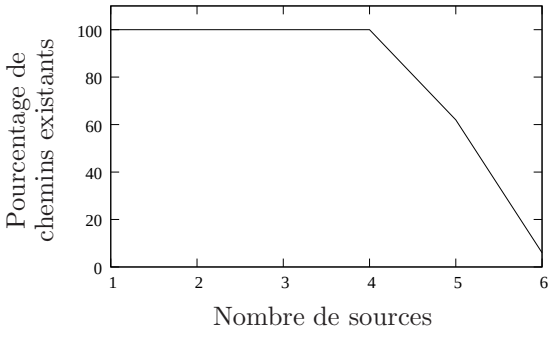


Figure C.3 – Pourcentage de topologies ayant une solution optimale en fonction du nombre de sources, avec un nombre de puits égal au nombre de sources.

Les simulations sont lancées sur un ordinateur personnel standard et prennent environ 2 h35 mn. Les résultats montrent que même sans avoir une limitation sur la capacité de calcul, des solutions optimales n'existent pas toujours, même sur de petits exemples. Dans un réseau de capteurs sans fil, où les nœuds ont une capacité de calcul limitée, il est nécessaire de disposer d'une heuristique simple qui est capable de fournir des solutions approximatives en utilisant des hypothèses réelles.

Liste des publications

1. **Nancy El Rachkidy**, Gérard Chalhoub, Alexandre Guitton et Michel Misson. « Queue-exchange Mechanism to Improve the QoS in a Multi-stack Architecture ». Dans *ACM PE-WASUN (Internation Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks)*. Novembre 2011. Prix du meilleur papier [ERCGM11]
2. **Nancy El Rachkidy**, Alexandre Guitton et Michel Misson. « Improving QoS in Wireless Sensor Networks using a Multi-stack Architecture ». Dans *IEEE VTC (Vehicular Technology Conference)*. Mai 2011. [ERGM11]
3. **Nancy El Rachkidy**, Alexandre Guitton, Bacem Bakhache et Michel Misson. « Address Assignment in Wireless Sensor Networks in Mines ». Dans *ICWCUCA (International Conference on Wireless Communications in Underground and Confined Areas)*. Août 2010. [ERGBM10]
4. **Nancy El Rachkidy**, Alexandre Guitton et Michel Misson. « Routing protocol for anycast communications in a wireless sensor network ». Dans *IFIP Networking*. Mai 2010. [ERGM10]
5. **Nancy El Rachkidy**, Alexandre Guitton et Michel Misson. « Optimizing the setup phase of an IEEE 802.15.4 wireless sensor network ». Dans *IFIP Wireless Days*. Décembre 2009. [ERGM09a]
6. **Nancy El Rachkidy**, Alexandre Guitton, Bacem Bakhache et Michel Misson. « PiRAT : Pivot Routing for Alarm Transmission in Wireless Sensor Networks ». Dans *IEEE LCN (Local Computer Networks)*. Octobre 2009. [ERGM09b]

Publications avant thèse :

1. Charles Yaacoub, Joumana Farah, **Nancy El Rachkidy** et Béatrice Pesquet-Popescu. « A Cross-Layer Approach for Dynamic Rate Allocation in H.264 Multi-User Video Streaming ». Dans *IEEE ICECS (International Conference on electronics, Circuits and Systems)*. Décembre 2007. [YFERP07a]
2. Charles Yaacoub, Joumana Farah, **Nancy El Rachkidy** et Béatrice Pesquet-Popescu. « Cross-Layer Optimization for Dynamic Rate Allocation in a Multi-User Video Streaming System ». Dans *IEEE NTMS (International Conference on New Technologies, Mobility and Security)*. Mai 2007. Poster. [YFERP07b]
3. Charles Yaacoub, Joumana Farah, **Nancy El Rachkidy** et Béatrice Pesquet-Popescu. « Dynamic RCPT-Based Cross-Layer Optimization for Multi-User H.264 Video Streaming ». Dans *ICENCO (International Computer Engineering Conference)*. Décembre 2006. Prix du meilleur papier [YFER$^+$06]
4. Joumana Farah, Charles Yaacoub, **Nancy El Rachkidy** et François Marx. « Binary and non-binary turbo codes for the compression of correlated sources transmitted through error-prone channels ». Dans *ISTC (ITG Conference on Source and Channel Coding)*. Avril 2006. [FYERM06]

Glossaire

Balise

Trame balise périodique utilisée pour délimiter un cycle et/ou définir une réference de temps dans un réseau. Un beacon peut éventuellement contenir des informations fonctionnelles.

densité du réseau

Nombre moyen de voisins (entités à porté) d'une entité réseau.

Détour

Passage d'un paquet plus d'une fois par une même entité réseau.

Latence

Temps nécessaire à un paquet de données pour passer de la source à la destination à travers un réseau.

Médium

Support par lequel les informations sont transmises

Portée

Champs de vision d'une entité réseau.

Puits

Entité réseau capable de stoquer les informations recoltées du réseau.

Qualité de service

Capacité à véhiculer dans de bonnes conditions un type de trafic donné, en termes de disponibilité, débit, délais de transmission, taux de pertes de trames.

Routage

Mécanisme par lequel des chemins sont sélectionnés dans un réseau pour acheminer les données d'un expéditeur jusqu'à un ou plusieurs destinataires.

Topologie

Architecture et organisation d'un réseau.

Trame

Paquet d'informations délimitées par des fanions.

Liste des abréviations

AMRT Accès Multiple à Répartition dans le Temps 39

ANR Agence Nationale de Recherche 76

AODV Ad hoc On-demand Distance Vector 33

BAAM Binary Address Assignement Mechanism 151

BE Backoff Exponent 27

BI Beacon Interval 26

CAP Contention Access Period 25

CFP Contention Free Period 25

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance 22

CSSS Closest Sink Selection Strategy 171

CW contention Window 27

DAAM Distributed Address Allocation Mechanism 35

DCNS Direction de Construction Navale Sextant 77

DPRD Distributed Passive Routing Desicions 44

EMPR Energy efficient MPR 86

Références

[AACG+09] K. Al Agha, G. Chalhoub, A. Guitton, E. Livolant, S. Mahfoudh, P. Minet, M. Misson, J. Rahmé, T. Val, and a. Van Den Bossche. Cross-layering in an industrial wireless sensor network : case study of OCARI. *Journal of Networks*, 4(6) :411–420, 2009.

[ACC09] A. Abbagnale, E. Cipollone, and F. Cuomo. A case study for evaluating IEEE 802.15.4 wireless sensor network formation with mobile sinks. Dans *IEEE ICC*, 2009.

[ANR] ANR. Agence Internationale de Recherche. La page web de ANR, http ://www.agence-nationale-recherche.fr.

[AVA06] I. F. Akyildiz, M. C. Vuran, and O. B. Akan. A Cross-Layer Protocol for Wireless Sensor Networks. Dans *Annual Conference on Information Sciences and Systems*, pages 1102–1107, 2006.

[AWW05] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks : a survey. *Computer networks*, 47 :445–487, 2005.

[All] ZigBee Alliance. The ZigBee Alliance website. http ://www.zigbee.org.

[Aso10] R. Asokan. A review of Quality of Service (QoS) routing protocols for mobile Ad hoc networks. Dans *International Conference on Wireless Communication and Sensor Computing, ICWCSC*, 2010.

[BBB09] F. Bouabdallah, N. Bouadballah, and R. Boutaba. Cross-Layer Design for Energy Conservation in Wireless Sensor Networks. Dans *IEEE ICC*, 2009.

[BGV05] C. Buratti, A. Giorgetti, and R. Verdone. Cross-layer design of an energy-efficient cluster formation algorithm with carrier-sensing multiple access for wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking archive*, 5(5) :672–685, 2005.

[BM04] K. Baldus, H. Klabunde and G. Muesch. Reliable set-up of medical body-sensor networks. Dans *European Workshop on Wireless Sensor Networks*, number 2920 in LNCS, Junuary 2004.

[BOV06] C. Buratti, J. Orris, and R. Verdone. On the Design of Tree-Based Topologies for Mutli-Sink Wireless Sensor Networks. septembre 2006.

[BPC+07] P. Baronti, P. Pillai, V. W. C. Chook, S. Chessa, A. Gotta, and Y. Fun Hu. Wireless sensor networks : A survey on the state of the art and the 802.15.4 and ZigBee standards. *Computer communications*, 30 :1655–1695, 2007.

[Bei08] D. Bein. Fault-tolerant k-fold Pivot Routing in Wireless Sensor Networks. Dans *41st Hawaii International Conference on System Sciences*, 2008.

[Bur01] A. Burr. *Modulation and Coding for Wireless Communications*. Pearson Education, 2001.

[CGH+02] E. Callaway, P. Gorday, L. Hester, J. Gutierrez, M. Naeve, B. Heile, and V. Bahl. Home networking with IEEE 802.15.4 : A developing standard for low-rate wireless personal area network. *IEEE Communications Magazine*, 40(8) :70–77, 2002.

[CGM08] G. Chalhoub, A. Guitton, and M. Misson. MAC specifications for a WPAN allowing both energy saving and guaranted delay - Part A : MaCARI : a synchronized tree-based MAC protocol. Dans *IFIP WSAN*, 2008.

[CHCP07] X. Carcelle, B. Heile, C. Chatellier, and P. Pailler. *Next WSN applications using ZigBee*, volume 256, pages 239–254. IFIP, 2007.

[CLG+09] G. Chalhoub, E. Livolant, A. Guitton, A. van den Bossche, M. Misson, and T. Val. Specifications and evaluation of a MAC protocol for a LP-WPAN. *Ad Hoc & Sensor Wireless Networks journal*, 2009.

[CT07] J. Chang and L. Tassiulas. Maximum Lifetime Routing in Wireless Sensor Networks. *IEEE/ACM Transactions on Networking*, 12(4), 2007.

[Dij59] E. W. Dijkstra. *A note on two problems in connexion with graphs.* Numerische mathematik 1, 269–271 edition, 1959.

[ERCGM11] N. El Rachkidy, G. Chalhoub, A. Guitton, and M. Misson. Queue-exchange Mechanism to Improve the QoS in a Multi-stack Architecture. Dans *ACM PE-WASUN*, novembre 2011.

[ERGBM10] N. El Rachkidy, A. Guitton, B. Bakhache, and M. Misson. Address assignment for wireless sensor networks in mines. Dans *ICWCUCA (International Conference on Wireless Communications in Underground and Confined Areas)*, août 2010.

[ERGM09a] N. El Rachkidy, A. Guitton, and M. Misson. Optimizing the setup phase of an IEEE 802.15.4 wireless sensor network. Dans *Wireless Days (IFIP Wireless Days)*, décembre 2009.

[ERGM09b] N. El Rachkidy, A. Guitton, and M Misson. Pirat : Pivot Routing for Alarm Transmission in Wireless Sensor Networks. Dans *IEEE Local Computer Networks*, 2009.

[ERGM10] N. El Rachkidy, A. Guitton, and M Misson. Routing protocol for anycast communications in a wireless sensor network. Dans *IFIP Networking*, LNCS, mai 2010.

[ERGM11] N. El Rachkidy, A. Guitton, and M. Misson. Improving QoS in Wireless Sensor Networks using a Multi-stack Architecture. Dans *VTC (IEEE Vehicular Technology Conference)*, mai 2011.

[FYERM06] J. Farah, C. Yaacoub, N. El Rachkidy, and F. Marx. Binary and non-binary turbo codes for the compression of correlated sources transmitted through error-prone channels. Dans *ISTC (ITG Conference on Source and Channel Coding)*, avril 2006.

[Fer05] D. et. al Ferrara. MACRO : An Integrated MAC/Routing Protocol for Geographical Forwarding in Wireless Sensor Networks. Dans *IEEE Infocom*, mars 2005.

[GKS03] S. Ganeriwal, R. Kumar, and M. B. Sribastava. Timing-sync protocol for sensor networks. Dans *ACM Embedded Networked Sensor Systems*, pages 138–149, 2003.

[Gav06] L. Gavrilovska. Cross-Layering Approaches in Wireless Ad Hoc Networks. *Wireless Personal Communications*, 37 :271–290, 2006.

[Gri98] R. Grimaldi. *Discrete and Combinatorial Mathematics : An Applied Introduction.* 4 edition, 1998.

[HGM09] N. Hadid, A. Guitton, and M. Misson. Adaptive slotted CSMA/CA algorithm for the traffic accumulated during the inactive period. Dans *ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor and Ubiquitous Networks*, 2009.

[Had11] N. Hadid. Utilisation d'un canal d'aparté pour des applications de réseaux de capteurs embarquées, 2011.

[ITU05] ITU. Propagation data and prediction method for the planning of indoor radio communication systems and local area networks in the frequency range of 900 MHz to 100 GHz. ITU, Recommendation ITU-R P 1238-4, 2005.

[JMS07] I. Jawhar, N. Mohamed, and K. Shuaib. A framework for pipeline infrastructure monitoring using wireless sensor networks. Dans *WTS (Annual Wireless Telecommunications Symposium)*, avril 2007.

[JS03] J. Jun and M. L. Sichitiu. The nominal capacity of wireless mech networks. *IEEE Wireless Comm. Mag.*, 10 :8–14, octobre 2003.

[KAAVN06] A. Koubaa, M. Alves, M. Attia, and A. Van Nieuwenhuyse. Collision-Free Beacon Scheduling Mechanisms for IEEE 802.15.4/Zigbee Cluster-Tree Wireless Sensor Networks. Technical Report TR-061104, Polytechnic Institute of Porto, novembre 2006.

[KKL+06] T. O. Kim, H. Kim, J. Lee, J. S. Park, and B. D. Choi. Performance analysis of IEEE 802.15.4 with non-beacon-enabled CSMA/CA in non-saturated condition. Dans *Embedded and Ubiquitous Computing*, volume 4096 of *LNCS*, pages 884–893, 2006.

[KKP+07] T. Kim, D. Kim, N. Park, S.-E. Yoo, and T. S. López. Shortcut tree routing in ZigBee networks. Dans *Proceedings IEEE Internation Symposium on Wireless Pervasive Computing (ISWPC)*, pages 42–47, février 2007.

[KR04] C. Kappler and C. Reigel. A real-world, simple wireless sensor network for monitoring electrical energy consumption. Dans *European Workshop on wireless Sensor Networks*, number 2920 in LNCS, Junuary 2004.

[KS06] M. Kalantari and M. Shayman. Design Optimization of Multi-sink Sensor Networks by analogy to Electrostatic Theory. Dans *IEEE WCNC*, 2006.

[KSCK05] H. Kim, Y. Seok, N. Choi, and T. Kwon. Optimal Multi-Sink Positioning and Energy-efficient Routing in Wireless Sensor Networks. Dans *Information Networking*, 2005.

[KUHN03] A. Kanzaki, T Uemukai, T. Hara, and S Nishio. Dynamic TDMA Slot Assignment in Ad Hoc Networks. Dans *17th International Conference on Advanced Information Networking and Applications (AINA)*, 2003.

[LHT+99] W. Lee, C. Henderson, H. F. Taylor, R. James, E. Lee, V. Swenson, R. A. Atkins, and W. G. Gemeiner. Railroad bridge instrumentation with fiber-optic sensors. *Applied Optics*, 38(7) :1110–1114, 1999.

[LJK07] J.S. Lee, Ryu J.P., and Han K.J. Efficient Routing Scheme Using Pivot Node in Wireless Sensor Networks. Dans *ICCS*, number 4490 in LNCS, pages 574–577, 2007.

[LZY08] D. Luo, D. Zuo, and X. Yang. An Optimal Sink Selection Scheme for Multi-sink Wireless Sensor Networks. Dans *Computer Science and Information Technology ICCSIT*, 2008.

[MFA07] G. Mao, B. Fidan, and B. D. O. Anderson. Wireless sensor network localization techniques. *Computer Networks*, 51(10) :2529–2553, juillet 2007.

[MM08a] S. Mahfoudh and P. Minet. EOLSR : an energy efficient routing protocol in wireless ad hoc and sensor networks. *Journal of Interconnection Networks*, 9(4), 2008.

[MM08b] S. Mahfoudh and P Minet. Survey of energy efficient strategies in wireless ad hoc and sensor networks. Dans *IEEE ICN*, 2008.

[MM09] S. Mahfoudh and P. Minet. Maximization of energy efficiency in wireless ad hoc and sensor networks with SERENA. *Journal of Mobile Information System (MIS)*, 2009.

[MMG10] P. Minet, G. Mahfoudh, S. Chalhoub, and A. Guitton. Node coloring in a wireless sensor network with unidirectional links and topology changes. Dans *IEEE WCNC*, 2010.

[MML09] M. V. Machado, R. Mini, and A. Loureiro. A Combined Approach for Receiver-based MAC and Network Layers in Wireless Sensor Networks. Dans *IEEE ICC*, 2009.

[Mah09] S. Mahfoudh. Energy efficiency in wireless ad hoc and sensor networks : routing, node activity scheduling and cross-layering, janvier 2009. Thèse à l'Université Pierre et Marie Curie.

[McK66] E. H. McKinney. Generalized Birthday Problem. *American Mathematical Monthly*, 73 :385–387, 1966.

[NS202] NS2. Network Simulator 2, 2002. http ://www.isi.edu.nsnam/ns.

[OE04] E. I. Oyman and C. Ersoy. Multiple sink Network Design Problem in LargeScale Wireless Sensor Networks. Dans *IEEE ICC*, 2004.

[PRH02] C. A. Pomalaza-Raez and T. L. Hemminger. A unified approach to dynamic TDMA slot assignment and to distributed routing for multi-hop packet radio networks. Dans *Artificial Neural Networks in Engineering (ANNIE)*, pages 975–980, 2002.

[PRML06] M. Petrova, J. Riihijarvi, P. Mahonen, and S. Labella. Performance study of IEEE 802.15.4 using measurements and simulations. Dans *IEEE WCNC*, pages 487–492, 2006.

[QPSS94] G. Qian-Ping, O. Satoshi, and P. Shietung. Efficient Algorithms for Node Disjoint Path Problems. *Proceedings of Electronics, Information and Communication Engineers Conference*, 2, 1994.

[RFC 3561] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-demand Distance Vector (AODV) Routing. Request For Comments 3561, IETF, juillet 2003.

[RFC 3626] C. Adjih, T. Clausen, P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot. Optimized Link State Routing Protocol. RFC 3626, IETF, 2003.

[RI04] V.T. Raisinghani and S. Iyer. Cross-layer Design Optimizations in Wireless Protocol Stacks. *Computer Communications*, 27, 2004.

[RWAS08] I. Rhee, A. Warrier, J. Aia, M. Min, and M.L. Sichitiu. Z-MAC : a hybrid MAC for wireless sensor networks. Dans *IEEE/ACM Transactions on Networking*, pages 511–524, 2008.

[SAB04] P. Skraba, H. Aghajan, and A. Bahai. Cross-layer optimization for high density sensor networks : Distributed passive routing Decisions. Dans *Ad-hoc Now*, juillet 2004.

[SL06] W. Su and T. L. Lim. Cross-Layer Design and Optimization for Wireless Sensor Networks. Dans *International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pages 278–284, 2006.

[SM04] A. Simon, G. Ledezezi and M. Maroti. Sensor network-based countersniper system. Dans *Sensys*, novembre 2004.

[SM05] I. Shukla and N. Menghanathan. Impact of Leader Selection Strategies on the PEGASIS Data Gathering Protocol for Wireless Sensor Networks. *Ubiquitous Computing and Communication Journal*, 2005.

[SM07] L. Stoianov, I. Nachman and S. Madden. Pipenet : A Wireless Sensor Network for Pipeline Monitoring. Dans *ACM IPSN*, 2007.

[SRDV08] P. Suarez, C.-G. Renmarker, A. Dunkels, and T. Voigt. Increasing ZigBee Network Lifetime with X-MAC. Dans *ACM RealWSN*, 2008.

[Sic04] M.L. Sichitiu. Cross-Layer Scheduling for Power Effeciency in Wireless Sensor Networks. Dans *IEEE Infocom*, 2004.

[TCC09] F. Turati, M. Cesana, and L. Campelli. SPARE MAC Enhanced : A Dynamic TDMA Protocol for Wireless Sensor Networks. Dans *IEEE Globecom*, 2009.

[WBS09] Z. Wang, E. Bulut, and B. K. Szymanski. Energy Efficient Collision Aware Multipath Routing for Wirelss Sensor Networks. Dans *IEEE ICC*, 2009.

[YFERP07a] C. Yaacoub, J. Farah, N. El Rachkidy, and B. PesquetPopescu. A cross-layer approach for dynamic rate allocation in h.264 multi-user video streaming. Dans *ICECS (IEEE International Conference on Electronics, Circuits and Systems)*, décembre 2007.

[YFERP07b] C. Yaacoub, J. Farah, N. El Rachkidy, and B. PesquetPopescu. Cross-layer optimization for dynamic rate allocation in multi-user video streaming system. Dans *NTMS (International conference on New Technologies, Mobility and Security)*, mai 2007. poster.

[YFER+06] C. Yaacoub, J. Farah, N. El Rachkidy, F. Marx, and B. PesquetPopescu. Dynamic RCPT-based cross-layer optimization for multi-user H.264 video streaming. Dans *ICENCO (International Computer Engineering Conference)*, décembre 2006.

[ZL04] J. Zheng and M. J. Lee. Will IEEE 802.15.4 make ubiquitous network a reality ? : A discussion on a potential low power, low bit rate standard. *IEEE Communications Magazine*, 27(6) :23–29, 2004.

[ZR03a] M. Zorzi and R. Rao. Geographic random forwarding (GeRaF) for ad hoc and sensor networks : energy and latency performance. *IEEE Trans. Mobile Computing*, 2(4) :349–365, 2003.

[ZR03b] M. Zorzi and R. Rao. Geographic random forwarding (GeRaF) for ad hoc and sensor networks : multihop performance. *IEEE Trans. Mobile Computing*, 2(4) :337–348, 2003.

[Zig08] ZigBee. ZigBee Specification. Standard Zigbee 053474r17, ZigBee Standards Organization, janvier 2008.

[IEE06] IEEE 802.15. Part 15.4 : Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). Standard 802.15.4 R2006, ANSI/IEEE, 2006.

[IEE07] IEEE 802.15. IEEE Standard for Information Technology Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks Specific Requirement Part 15.4 : Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). Standard 802.15.4a, ANSI/IEEE, 2007.

Résumé

La tendance actuelle des réseaux de capteurs sans fil est d'avoir un seul réseau supportant plusieurs applications et fournissant plusieurs QoS. Dans cette thèse, nous étudions les techniques de *cross-layering* afin d'améliorer les performances et de fournir de la QoS.

Tout d'abord, nous généralisons le concept de la méthode d'accès MaCARI en proposant une architecture multi-couches où plusieurs combinaisons de protocoles MAC-routage sont utilisées. Une file d'attente est associée à chaque combinaison, et chaque combinaison est activée pour une période précise. Le but est de profiter de ces combinaisons pour offrir différentes QoS. Cependant, cette architecture cause un problème de dimensionnenemt des périodes, ce qui a un impact sur les performances du réseau. Nous proposons, ensuite, des techniques de *cross-layering* en échangeant les paquets entre les différentes files d'attente afin de résoudre le problème de dimensionnement. Durant sa période, chaque combinaison traite tous les paquets de sa file d'attente ainsi que les paquets des files d'attente d'autres période. Nous montrons par simulation que notre approche améliore les performances du réseau.
Mots clés : Réseaux de capteurs sans fil, méthode d'accès au médium, protocoles de routage, QoS, *cross-layering*.

Abstract

The current trend in wireless sensor networks is to have a single network supporting serveral applications and providing several QoS. In this thesis, we study the cross-layering techniques in order to improve the network performance and provide several QoS.

Fistly, we generalize the concept of the access method MaCARI by proposing a multi-stack architecture in which several MAC-routing combination protocols are used. A queue is associated to each combination, and each combination is active for a specified period. The purpose consists in using these combinations in order to provide different QoS. However, this architecture yields to a dimensioning problem for the periods reducing the network performance. Secondly, we propose cross-layering techniques by exchanging packets between different queues to solve the dimensioning problem. During its period, each combination treats all the packets of its queue and the packets related to queues associated to other periods. We show by simulations that our approach improves the network performance.
Keywords : Wireless sensor networks, MAC, routing protocols, QoS, cross-layering.